96

10/626,608

日本国特許庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２００３年　７月２２日

出願番号
Application Number: 特願２００３－１９９９４７

[ST. 10/C]: [ＪＰ２００３－１９９９４７]

出願人
Applicant(s): 株式会社リコー

２００３年　８月　４日

特許庁長官
Commissioner,
Japan Patent Office 今井康夫

出証番号　出証特２００３－３０６２１７０

【書類名】　特許願
【整理番号】　0305118
【提出日】　平成15年 7月22日
【あて先】　特許庁長官　今井　康夫　殿
【国際特許分類】　G03G 21/00　370
【発明の名称】　画像形成装置およびアプリケーション実行方法
【請求項の数】　31
【発明者】
　【住所又は居所】　東京都大田区中馬込１丁目３番６号　株式会社リコー内
　【氏名】　大石　勉
【発明者】
　【住所又は居所】　東京都大田区中馬込１丁目３番６号　株式会社リコー内
　【氏名】　杉浦　裕子
【特許出願人】
　【識別番号】　000006747
　【氏名又は名称】　株式会社リコー
【代理人】
　【識別番号】　100070150
　【弁理士】
　【氏名又は名称】　伊東　忠彦
【先の出願に基づく優先権主張】
　【出願番号】　特願2002-218814
　【出願日】　平成14年 7月26日
【先の出願に基づく優先権主張】
　【出願番号】　特願2002-295378
　【出願日】　平成14年10月 8日
【手数料の表示】
　【予納台帳番号】　002989
　【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　9911477

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　画像形成装置およびアプリケーション実行方法

【特許請求の範囲】

【請求項１】　画像形成処理に関するシステム側の処理を行うサービスモジュールを有し、当該サービスモジュールとは別にアプリケーションを搭載可能に構成された画像形成装置において、

アプリケーションを実行する仮想マシンと、

当該仮想マシンにより実行されるアプリケーションを管理するアプリケーション管理部と

を備えたことを特徴とする画像形成装置。

【請求項２】　前記仮想マシンに代えてインタプリタを備えた請求項１に記載の画像形成装置。

【請求項３】　前記アプリケーションはＪａｖａ（登録商標）プログラムであり、前記画像形成装置は、当該Ｊａｖａ（登録商標）プログラムが参照するクラスライブラリを備えた請求項１に記載の画像形成装置。

【請求項４】　前記クラスライブラリは、前記画像形成装置の表示部への画面表示を行うためのクラスを含む請求項３に記載の画像形成装置。

【請求項５】　前記アプリケーション管理部は、前記画像形成装置に接続される記憶媒体から、又はネットワークを介して前記画像形成装置に接続されるサーバから、アプリケーションを前記画像形成装置にロードするローダーである請求項１又は２に記載の画像形成装置。

【請求項６】　前記ローダーは、アプリケーションが前記画像形成装置にロードされているか否かを判断し、ロードされていれば当該アプリケーションを実行し、ロードされていなければローダーの画面を表示する請求項１又は２に記載の画像形成装置。

【請求項７】　前記ローダーは、アプリケーションロード画面を表示し、ユーザにより指定された場所からアプリケーションをダウンロードする請求項１又は２に記載の画像形成装置。

【請求項８】　前記ローダーは、アプリケーションのリストを表示し、ユー

ザにより選択されたアプリケーションをダウンロードする請求項７に記載の画像形成装置。

【請求項９】　前記場所は、アプリケーションを格納するＷｅｂサーバ又はＦＴＰサーバである請求項７に記載の画像形成装置。

【請求項１０】　前記ローダーは、ダウンロードしたアプリケーションが、前記場所において更新されているか否かを所定の間隔でチェックする請求項７に記載の画像形成装置。

【請求項１１】　前記ローダーは、前記所定の間隔を入力するための画面を表示し、ユーザからの指示に基づき当該所定の間隔を設定する請求項１０に記載の画像形成装置。

【請求項１２】　アプリケーションを連結して、連結されたアプリケーションにより一連の処理を実行させる連結手段を更に備えた請求項１又は２に記載の画像形成装置。

【請求項１３】　前記連結手段は、所定のアプリケーションと、その所定のアプリケーションに接続される候補となる複数のアプリケーションを示す画面を表示し、当該複数のアプリケーションの中からユーザにより選択されたアプリケーションを前記所定のアプリケーションに接続する請求項１２に記載の画像形成装置。

【請求項１４】　画像形成処理に関するシステム側の処理を行うサービスモジュールを有し、当該サービスモジュールとは別にアプリケーションを搭載可能に構成された画像形成装置を、

アプリケーションを実行する手段としての仮想マシン、

当該仮想マシンにより実行されるアプリケーションを管理するアプリケーション管理部

として機能させるプログラム。

【請求項１５】　画像形成処理に関するシステム側の処理を行うサービスモジュールを有し、当該サービスモジュールとは別にアプリケーションを搭載可能に構成された画像形成装置を、

アプリケーションを実行する手段としてのインタプリタ、

当該インタプリタにより実行されるアプリケーションを管理するアプリケーション管理部

として機能させるプログラム。

【請求項１６】　前記アプリケーションはＪａｖａ（登録商標）プログラムであり、前記プログラムは、当該Ｊａｖａ（登録商標）プログラムが参照するクラスライブラリを含む請求項１４に記載のプログラム。

【請求項１７】　前記クラスライブラリは、前記画像形成装置の表示部への画面表示を行うためのクラスを含む請求項１６に記載のプログラム。

【請求項１８】　前記アプリケーション管理部は、前記画像形成装置に接続される記憶媒体から、又はネットワークを介して前記画像形成装置に接続されるサーバから、アプリケーションを前記画像形成装置にロードするローダーである請求項１４又は１５に記載のプログラム。

【請求項１９】　前記ローダーは、アプリケーションが前記画像形成装置にロードされているか否かを判断し、ロードされていれば当該アプリケーションを実行し、ロードされていなければローダーの画面を表示する請求項１４又は１５に記載のプログラム。

【請求項２０】　前記ローダーは、アプリケーションロード画面を表示し、ユーザにより指定された場所からアプリケーションをダウンロードする請求項１４又は１５に記載のプログラム。

【請求項２１】　前記ローダーは、アプリケーションのリストを表示し、ユーザにより選択されたアプリケーションをダウンロードする請求項２０に記載のプログラム。

【請求項２２】　前記場所は、アプリケーションを格納するＷｅｂサーバ又はＦＴＰサーバである請求項２０に記載のプログラム。

【請求項２３】　前記ローダーは、ダウンロードしたアプリケーションが、前記場所において更新されているか否かを所定の間隔でチェックする請求項２０に記載のプログラム。

【請求項２４】　前記ローダーは、前記所定の間隔を入力するための画面を表示し、ユーザからの指示に基づき当該所定の間隔を設定する請求項２３に記載

のプログラム。

【請求項２５】　前記画像形成装置を、アプリケーションを連結して、連結されたアプリケーションにより一連の処理を実行させる連結手段として更に機能させる請求項１４又は１５に記載のプログラム。

【請求項２６】　前記連結手段は、所定のアプリケーションと、その所定のアプリケーションに接続される候補となる複数のアプリケーションを示す画面を表示し、当該複数のアプリケーションの中からユーザにより選択されたアプリケーションを前記所定のアプリケーションに接続する請求項２５に記載のプログラム。

【請求項２７】　請求項１４ないし２６のうちいずれか１項に記載のプログラムを記録したコンピュータ読み取り可能な記録媒体。

【請求項２８】　アプリケーションを実行する仮想マシンと、当該仮想マシンにより実行されるアプリケーションを管理するアプリケーション管理部とを備えた画像形成装置におけるアプリケーション実行方法であって、

前記アプリケーション管理部が、アプリケーションロード画面を表示し、ユーザにより指定された場所からアプリケーションをダウンロードし、ダウンロードしたアプリケーションを実行するステップを有することを特徴とするアプリケーション実行方法。

【請求項２９】　前記画像形成装置は、前記仮想マシンに代えてインタプリタを備えた請求項２８に記載のアプリケーション実行方法。

【請求項３０】　前記場所は、前記画像形成装置に接続される記録媒体、又はネットワークを介して前記画像形成装置に接続されるサーバである請求項２８又は２９に記載のアプリケーション実行方法。

【請求項３１】　前記アプリケーション管理部は、アプリケーションのリストを表示し、ユーザにより選択されたアプリケーションをダウンロードする請求項２８又は２９に記載のアプリケーション実行方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

この発明は、アプリケーションを容易に実行することができる画像形成装置およびアプリケーション実行方法に関するものである。

【従来の技術】

近年では、プリンタ、コピー、ファクシミリ、スキャナなどの各装置の機能を１つの筐体内に収納した画像形成装置（以下、「複合機」という。）が知られている。この複合機は、１つの筐体内に表示部、印刷部および撮像部などを設けるとともに、プリンタ、コピーおよびファクシミリ装置にそれぞれ対応した３種類のソフトウェアを設け、これらのソフトウェアを切り替えることによって、当該装置をプリンタ、コピー、スキャナ又はファクシミリ装置として動作させるものである。

【０００２】

このような従来の複合機では、利用者に対し複合機の各種操作を行わせるためのオペレーションパネルが設けられ、このオペレーションパネルに操作画面の表示およびタッチ操作入力を行わせるための操作表示部を表示している。従来の複合機では、予め提供される機能が定められていたため、利用者が複合機に対して行う操作も大きな変更はなく、操作表示部の画面表示やタッチ操作による動作をカスタマイズする必要性はない。このため、従来の複合機には操作表示部のカスタマイズ機能は搭載されていなかった。

【０００３】

【特許文献１】

特開２００２－１５２４４６号公報

【０００４】

【発明が解決しようとする課題】

ところで、このような従来の複合機では、プリンタ、コピー、スキャナおよびファクシミリ装置に対応するソフトウェアをそれぞれ別個に設けているため、各ソフトウェアの開発に多大の時間を要する。このため、出願人は、表示部、印刷部および撮像部などの画像形成処理で使用されるハードウェア資源を有し、プリンタ、コピー又はファクシミリなどの各ユーザサービスにそれぞれ固有の処理を行うアプリケーションを複数搭載し、これらのアプリケーションとハードウェア

資源との間に介在して、ユーザサービスを提供する際に、アプリケーションの少なくとも２つが共通に必要とするハードウェア資源の管理、実行制御並びに画像形成処理を行う各種コントロールサービスからなるプラットホームを備えた画像形成装置（複合機）を発明した。なお、各種コントロールサービスが行う処理をシステム側の処理と呼ぶ。

【０００５】

このような新規な複合機では、アプリケーションと、ハードウェア資源にアクセスするような開発が難しい処理を行うコントロールサービスとを別個に設けているため、複合機の出荷後にユーザもしくは第三者であるサードベンダが画像形成処理などにかかるアプリケーションとして新規な外部アプリケーションを開発して複合機に搭載可能な構成となっている。

【０００６】

このため、新たに開発された外部アプリケーションの起動や実行中に操作表示部における画面表示やタッチ操作によって、予め提供されている機能とは異なる実行処理を行わせたい場合がある。このように、新規な複合機では、操作表示部のカスタマイズという、出荷後に外部アプリケーションを搭載することを想定していない従来の複合機では問題にならなかった新規な課題が生じてくる。

【０００７】

このような操作表示部のカスタマイズを、外部アプリケーションの開発段階でソースコードを記述し、ソースコードをコンパイルおよびリンクすることによって行うことが考えられる。すなわち、外部アプリケーションの開発段階で、外部アプリケーション自体の処理の中に操作表示部のタッチ操作に伴う実行処理を開発する場合が多くなる。この場合、操作表示部のタッチ操作に伴う実行処理の検証を逐次行いながら外部アプリケーション自体の開発を進めていくことがプログラム開発効率上好ましい。しかしながら、タッチ操作に伴う実行処理に障害を検出するたびに、ソースコードの修正、再コンパイル、再リンクを行って、開発対象の複合機に再インストールし、動作検証を行わなければならないとすると、プログラム開発の効率が悪くなるという問題がある。

【０００８】

また、操作表示部のカスタマイズに限らず、複合機とは別のＰＣ等の環境でのプログラムの作成と、複合機での動作検証とによる従来のプログラム開発は効率が悪いという問題がある。

【０００９】

更に、効率よくアプリケーションプログラムを開発することができたとしても、当該プログラムを複合機に格納して実行するのに手間がかかるようでは利便性に問題がある。

【００１０】

この発明は上記に鑑みてなされたもので、アプリケーションを容易に実行可能な画像形成装置およびアプリケーション実行方法を得ることを目的とする。

【課題を解決するための手段】

上記の課題は以下の発明により解決される。

【００１１】

請求項１に記載の発明は、画像形成処理に関するシステム側の処理を行うサービスモジュールを有し、当該サービスモジュールとは別にアプリケーションを搭載可能に構成された画像形成装置であり、アプリケーションを実行する仮想マシンと、当該仮想マシンにより実行されるアプリケーションを管理するアプリケーション管理部とを備えたものである。

【００１２】

本発明によれば、仮想マシン上で動作するアプリケーションを画像形成装置で実行できる。一般に仮想マシンは機種の違いを吸収するので、アプリケーションを容易に画像形成装置上で実行できる。

【００１３】

請求項２に記載の発明は、前記仮想マシンに代えてインタプリタを備えたものであり、請求項１の発明と同様の効果を奏する。

【００１４】

請求項３に記載の発明は、請求項１の記載において、前記アプリケーションはＪａｖａ（登録商標）プログラムであり、前記画像形成装置は、当該Ｊａｖａ（登録商標）プログラムが参照するクラスライブラリを備えるものである。

【００１５】

本発明によれば、クラスライブラリの提供する種々の機能を利用したアプリケーションを実行できる。

【００１６】

請求項４に記載の発明は、請求項３の記載において、前記クラスライブラリは、前記画像形成装置の表示部への画面表示を行うためのクラスを含むものである。

【００１７】

請求項５に記載の発明は、請求項１又は２の記載において、前記アプリケーション管理部は、前記画像形成装置に接続される記憶媒体から、又はネットワークを介して前記画像形成装置に接続されるサーバから、アプリケーションを前記画像形成装置にロードするローダーである。

【００１８】

本発明によれば、ローダーがアプリケーションをロードするので、容易にアプリケーションを実行できる。

【００１９】

請求項６に記載の発明は、請求項１又は２の記載において、前記ローダーは、アプリケーションが前記画像形成装置にロードされているか否かを判断し、ロードされていれば当該アプリケーションを実行し、ロードされていなければローダーの画面を表示するものである。

【００２０】

本発明によれば、アプリケーションを容易に実行できる。また、アプリケーションが存在しない場合には、ローダーの画面が表示されるので、容易にアプリケーションをロードできる。

【００２１】

請求項７に記載の発明は、請求項１又は２の記載において、前記ローダーは、アプリケーションロード画面を表示し、ユーザにより指定された場所からアプリケーションをダウンロードするものである。

【００２２】

請求項８に記載の発明は、請求項７の記載において、前記ローダーは、アプリケーションのリストを表示し、ユーザにより選択されたアプリケーションをダウンロードする。

【００２３】

本発明によっても容易にアプリケーションをロードでき、実行できる。

【００２４】

請求項９に記載の発明は、請求項７の記載において、前記場所は、アプリケーションを格納するＷｅｂサーバ又はＦＴＰサーバであるとするものである。

【００２５】

本発明により、ＵＲＬを指定するだけでアプリケーションをダウンロードできる。

【００２６】

請求項１０に記載の発明は、請求項７の記載において、前記ローダーは、ダウンロードしたアプリケーションが、前記場所において更新されているか否かを所定の間隔でチェックするものである。

【００２７】

本発明によれば、バージョンアップしたアプリケーションを容易に取得できる。

【００２８】

請求項１１に記載の発明は、請求項１０の記載において、前記ローダーは、前記所定の間隔を入力するための画面を表示し、ユーザからの指示に基づき当該所定の間隔を設定する。これにより間隔を設定できる。

【００２９】

請求項１２に記載の発明は、請求項１又は２の記載において、アプリケーションを連結して、連結されたアプリケーションにより一連の処理を実行させる連結手段を更に備えるものである。

【００３０】

本発明により、アプリケーション実行のカスタマイズを容易に行うことが可能となる。

【００３１】

請求項１３に記載の発明は、請求項１２の記載において、前記連結手段は、所定のアプリケーションと、その所定のアプリケーションに接続される候補となる複数のアプリケーションを示す画面を表示し、当該複数のアプリケーションの中からユーザにより選択されたアプリケーションを前記所定のアプリケーションに接続するものである。

【００３２】

本発明により、ユーザは視覚的にアプリケーション実行のカスタマイズを行うことが可能となる。

【００３３】

請求項１４～２６に記載の発明は、上記画像形成装置の機能を実現するためのプログラムの発明である。請求項２７に記載の発明は、そのプログラムを記録した記録媒体である。この発明によっても画像形成装置の発明と同様の効果を奏する。また、請求項２８～３１に記載の発明は、ローダーによりアプリケーションをダウンロードして実行するアプリケーション実行方法の発明である。この発明によっても画像形成装置の発明と同様の効果を奏する。

【００３４】

【発明の実施の形態】

以下に添付図面を参照して、この発明にかかる画像形成装置および操作表示部プログラム生成方法の好適な実施の形態を詳細に説明する。

【００３５】

（実施の形態１）

図１は、この発明の実施の形態１である画像形成装置（以下、「複合機」という）の主要部の構成とネットワーク構成を示すブロック図である。実施の形態１にかかる複合機１００は、インターネットに接続され、当該インターネットに接続されたＰＣ（Ｐｅｒｓｏｎａｌ　Ｃｏｍｐｕｔｅｒ）などのクライアント端末から複合機の操作を行うオペレーションパネルの操作表示部のカスタマイズプログラムの入力およびデバッグ作業を行うことを可能としたものである。

【００３６】

図１に示すように、複合機１００とＰＣ２００は、インターネット２２０で接続されており、通信プロトコルとしてＴＣＰ／ＩＰを利用している。複合機１００は、複合機初期化部１２９と、プログラム起動部１３１と、プログラミングサービス１３２と、インタプリタ１３４と、ＮＣＳ（ネットワークコントロールサービス）１２８と、ｈｔｔｐｄ（ｈｔｔｐデーモン）１０６と、共有メモリ１０５と、ＨＤＤ１０３とを備えた構成となっている。

【００３７】

プログラミングサービス１３２は、ネットワークに接続されたクライアント端末としてのＰＣ２００のＷＥＢブラウザに後述するｈｔｔｐファイルで構成されたＷＥＢページであるプログラミング画面２０１を表示し、このプログラミング画面２０１からオペレーションパネル２１０の操作表示部のカスタマイズプログラムを入力させるものである。また、プログラミングサービス１３２は、ＰＣから入力されたカスタマイズプログラムをＨＤＤ１０３のカスタマイズ領域として作成されたカスタマイズディレクトリ２１２に保存する。さらに、プログラミングサービス１３２は、複合機１００上でカスタマイズプログラムが実行されるように、後述する起動設定ファイル２１１に入力されたカスタマイズプログラムを登録する。

【００３８】

ここで、プログラミングサービス１３２とＰＣ２００のＷＥＢブラウザの間のプログラミング画面などのｈｔｔｐファイルの転送は、ｈｔｔｐプロトコルに従って行われる。このため、プログラミングサービス１３２は、Ｗｅｂサーバ（ｈｔｔｐ サーバ）としての役割も担っている。

【００３９】

カスタマイズプログラムは、インタプリタで解釈可能な言語で記述され、具体的には、汎用ＯＳがＵＮＩＸ（登録商標）の場合、シェルスクリプトである。

【００４０】

インタプリタ１３４は、カスタマイズプログラムをステップごとに解析し逐次実行するものであり、本実施の形態では、シェルスクリプトを実行するシェル（ｂｓｈ，ｃｓｈなど）である。このシェル１３４は、汎用ＯＳ１２１としてのＵ

ＮＩＸ（登録商標）にＯＳのカーネル１３５とは別個に存在するものである。

【００４１】

プログラム起動部１３１は、ＨＤＤ１０３の診断結果が正常である場合に、ＨＤＤ１０３にインストールされている外部アプリ１１７を起動するものである。また、プログラム起動部１３１は、ＩＣカードなどの記憶媒体に保存された外部アプリ１１７を起動する。

【００４２】

複合機初期化部１２９は、汎用ＯＳ１２１の上で最初に起動されるプロセスであり、コントロールサービスやアプリケーション１３０（外部アプリ１１７を除く）の起動およびプログラム起動部１３１の起動を行うものである。

【００４３】

ＮＣＳ１２８は、ネットワークを制御するものであり、ｈｔｔｐｄ１０６から通知されたリクエストメッセージ受信の旨をプログラミングサービス１３２に通知する。

【００４４】

ｈｔｔｐｄ１０６は、ＯＳに含まれるプロセス（デーモン）であり、ポート８０番を常時監視してリクエストメッセージの受信を行うとともに、レスポンスメッセージの送信を行うものである。なお、リクエストメッセージおよびレスポンスメッセージの構造は、通常のｈｔｔｐプロトコルにおける各メッセージの構造と同様であり、各メッセージには、ｈｔｍｌ形式で記述されたメッセージボディが含まれている。また、ｈｔｔｐｄ１０６は、インターネット２２０経由で受信したリクエストメッセージ受信の旨をＮＣＳ１２８に通知するとともに、リクエストメッセージを共有メモリ１０５ に格納する。

【００４５】

この共有メモリ１０５は、ｈｔｔｐｄ１０６とプログラミングサービス１３２とのプロセス間通信に利用されるものであり、共有メモリ１０５を介してリクエストメッセージ、レスポンスメッセージなどの受け渡しを行う。

【００４６】

ＨＤＤ１０３には、起動設定ファイル２１１が格納され、またＨＤＤ１０３の

カスタマイズディレクトリ２１２にカスタマイズプログラムが格納される。

【００４７】

ＰＣ２００は、複合機１００のプログラミングサービス１３２をサーバとしたクライアント端末であり、ＰＣ２００で実行されるＷＥＢブラウザにプログラミングサービス１３２からｈｔｔｐプロトコルで送信されたプログラミング画面２０１やその他画面を表示し、操作入力を行えるように構成されている。

【００４８】

次に、本実施の形態にかかる複合機１００の全体の機能的構成について説明する。図２は、実施の形態１の複合機１００の構成を示すブロック図である。図２に示すように、複合機１００は、白黒ラインプリンタ（B&W LP）１０１と、カラーラインプリンタ（Color LP）１０２と、ハードディスク装置（ＨＤＤ）１０３と、スキャナ、ファクシミリ、メモリ、ネットワークインタフェースなどのハードウェアリソース１０４を有するとともに、プラットホーム１２０と、アプリケーション１３０と、複合機初期化部１２９と、プログラム起動部１３１と、プログラミングサービス１３２とから構成されるソフトウェア群１１０とを備えている。

【００４９】

プラットホーム１２０は、アプリケーションからの処理要求を解釈してハードウェア資源の獲得要求を発生させるコントロールサービスと、一又は複数のハードウェア資源の管理を行い、コントロールサービスからの獲得要求を調停するシステムリソースマネージャ（ＳＲＭ）１２３と、汎用ＯＳ１２１とを有する。

【００５０】

コントロールサービスは、複数のサービスモジュールから形成され、ＳＣＳ（システムコントロールサービス）１２２と、ＥＣＳ（エンジンコントロールサービス）１２４と、ＭＣＳ（メモリコントロールサービス）１２５と、ＯＣＳ（オペレーションパネルコントロールサービス）１２６と、ＦＣＳ（ファックスコントロールサービス）１２７と、ＮＣＳ（ネットワークコントロールサービス）１２８とから構成される。また、プログラミングサービス１３２も、コントロールサービス層に含まれている。なお、このプラットホーム１２０は、あらかじめ定

義された関数により前記アプリケーション１３０から処理要求を受信可能とするアプリケーションプログラムインタフェース（ＡＰＩ）を有する。

【００５１】

汎用ＯＳ１２１は、ＵＮＩＸ（登録商標）などの汎用オペレーティングシステムであり、プラットホーム１２０並びにアプリケーション１３０の各ソフトウェアをそれぞれプロセスとして並列実行する。

【００５２】

ＳＲＭ１２３のプロセスは、ＳＣＳ１２２とともにシステムの制御およびリソースの管理を行うものである。ＳＲＭ１２３のプロセスは、スキャナ部やプリンタ部などのエンジン、メモリ、ＨＤＤファイル、ホストＩ／Ｏ（セントロＩ／Ｆ、ネットワークＩ／Ｆ 、ＩＥＥＥ１３９４ Ｉ ／Ｆ、ＲＳ２３２Ｃ Ｉ／Ｆなど）のハードウェア資源を利用する上位層からの要求にしたがって調停を行い、実行制御する。

【００５３】

具体的には、このＳＲＭ１２３は、要求されたハードウェア資源が利用可能であるか（他の要求により利用されていないかどうか）を判断し、利用可能であれば要求されたハードウェア資源が利用可能である旨を上位層に伝える。また、ＳＲＭ１２３は、上位層からの要求に対してハードウェア資源の利用スケジューリングを行い、要求内容（例えば、プリンタエンジンにより紙搬送と作像動作、メモリ確保、ファイル生成など）を直接実施している。

【００５４】

ＳＣＳ１２２のプロセスは、アプリ管理、操作部制御、システム画面表示、ＬＥＤ表示、リソース管理、割り込みアプリ制御などを行う。

【００５５】

ＥＣＳ１２４のプロセスは、白黒ラインプリンタ（B&W LP）１０１、カラーラインプリンタ（Color LP）１０２、スキャナ、ファクシミリなどからなるハードウェアリソース１０３のエンジンの制御を行う。

【００５６】

ＭＣＳ１２５のプロセスは、画像メモリの取得および解放、ハードディスク装

置（ＨＤＤ）の利用、画像データの圧縮および伸張などを行う。

【００５７】

ＦＣＳ１２７のプロセスは、システムコントローラの各アプリ層からＰＳＴＮ／ＩＳＤＮ 網を利用したファクシミリ送受信、ＢＫＭ （バックアップＳＲＡＭ）で管理されている各種ファクシミリデータの登録／引用、ファクシミリ読みとり、ファクシミリ受信印刷、融合送受信を行うためのＡＰＩを提供する。

【００５８】

ＮＣＳ１２８のプロセスは、ネットワークＩ／Ｏ を必要とするアプリケーションに対して共通に利用できるサービスを提供するためのプロセスであり、ネットワーク側から各プロトコルによって受信したデータを各アプリケーションに振り分けたり、アプリケーションからデータをネットワーク側に送信する際の仲介を行う。具体的には、ftpd、lpd、snmpd、telnetd、smtpdなどのサーバデーモンや、同プロトコルのクライアント機能などを有している。

【００５９】

ＯＣＳ１２６のプロセスは、オペレータ（ユーザ）と本体制御間の情報伝達手段となるオペレーションパネル（操作パネル）２１０の制御を行う。ＯＣＳ１２６は、オペレーションパネルからキー押下（又はタッチ操作）をキーイベントとして取得し、取得したキーに対応したキーイベント関数をＳＣＳ１２２に送信するＯＣＳプロセスの部分と、アプリケーション１３０又はコントロールサービスからの要求によりオペレーションパネルに各種画面を描画出力する描画関数やその他オペレーションパネルに対する制御を行う関数などがあらかじめ登録されたＯＣＳライブラリの部分とから構成される。このＯＣＳライブラリは、アプリケーション１３０およびコントロールサービスの各モジュールにリンクされて実装されている。なお、ＯＣＳ１２６のすべてをプロセスとして動作させるように構成しても良く、あるいはＯＣＳ１２６のすべてをＯＣＳライブラリとして構成しても良い。

【００６０】

アプリケーション１３０は、ページ記述言語（ＰＤＬ）、ＰＣＬおよびポストスクリプト（ＰＳ）を有するプリンタ用のアプリケーションであるプリンタアプ

リ１１１と、コピー用アプリケーションであるコピーアプリ１１２と、ファクシミリ用アプリケーションであるファックスアプリ１１３と、スキャナ用アプリケーションであるスキャナアプリ１１４と、ネットワークファイル用アプリケーションであるネットファイルアプリ１１５と、工程検査用アプリケーションである工程検査アプリ１１６　とを有している。

【００６１】

アプリケーション１３０の各プロセス、コントロールサービスの各プロセスは、関数呼び出しとその戻り値送信およびメッセージの送受信によってプロセス間通信を行いながら、コピー、プリンタ、スキャナ、ファクシミリなどの画像形成処理にかかるユーザサービスを実現している。

【００６２】

このように、実施の形態１にかかる複合機１００には、複数のアプリケーション１３０および複数のコントロールサービスが存在し、いずれもプロセスとして動作している。そして、これらの各プロセス内部には、一又は複数のスレッドが生成されて、スレッド単位の並列実行が行われる。そして、コントロールサービスがアプリケーション１３０に対し共通サービスを提供しており、このため、これらの多数のプロセスが並列動作、およびスレッドの並列動作を行って互いにプロセス間通信を行って協調動作をしながら、コピー、プリンタ、スキャナ、ファクシミリなどの画像形成処理にかかるユーザサービスを提供するようになっている。

【００６３】

また、複合機１００には、複合機１００の顧客、サードベンダなどの第三者がコントロールサービス層の上のアプリケーション層に外部アプリを開発して搭載することが可能となっている。

【００６４】

なお、実施の形態１にかかる複合機１００では、複数のアプリケーション１３０のプロセスと複数のコントロールサービスのプロセスとが動作しているが、アプリケーション１３０とコントロールサービスのプロセスがそれぞれ単一の構成とすることも可能である。また、各アプリケーション１３０は、アプリケーショ

ンごとに追加又は削除することができる。

【００６５】

また、複合機１００はインストーラを含む。インストーラは、第三者が開発した外部アプリ１１７を、ＨＤＤ１０３にインストールするものである。本実施の形態にかかる複合機１００では、フラッシュカードなどのＩＣカードの記憶媒体に外部アプリ１１７を格納し、インストーラによって外部アプリ１１７をＨＤＤ１０３にインストールし、プログラム起動部１３１によって、ＨＤＤ１０３から外部アプリ１１７を起動してアプリケーション層で動作させるようになっている。一方、プリンタアプリ１１１、コピーアプリ１１２、ファックスアプリ１１３、スキャナアプリ１１４、ネットファイルアプリ１１５、工程検査アプリ１１６などの複合機１００の出荷時に提供されるアプリケーション１３０、各コントロールサービス、ＨＤＤ診断部１３２およびプログラム起動部１３１は、フラッシュメモリに出荷時に組み込まれており、複合機１００の起動時（電源投入時）に複合機初期化部１２９によって起動されるようになっている。

【００６６】

図３に複合機１００のハードウェア構成例を示す。

【００６７】

複合機１００は、コントローラ１６０と、オペレーションパネル１７５と、ファックスコントロールユニット（ＦＣＵ）１７６と、プリンタ等の画像形成処理に特有のハードウェア資源であるエンジン部１７７とを含む。コントローラ１６０は、ＣＰＵ１６１と、システムメモリ１６２と、ノースブリッジ（ＮＢ）１６３と、サウスブリッジ（ＳＢ）１６４と、ＡＳＩＣ１６６と、ローカルメモリ１６７と、ＨＤＤ１６８と、ネットワークインターフェースカード（ＮＩＣ）１６９と、ＳＤカード用スロット１７０と、ＵＳＢデバイス１７１と、ＩＥＥＥ１３９４デバイス１７２と、セントロニクス１７３とを含む。なお、メモリ１６２、１６７はＲＡＭ、ＲＯＭ等を含む。ＦＣＵ１７６およびエンジン部１７７は、コントローラ１６０のＡＳＩＣ１６６にＰＣＩバス１７８で接続されている。

【００６８】

ＣＰＵ１６１が、複合機１００にインストールされるアプリケーション、コン

トロールサービス等のプログラムを、メモリから読み出して実行する。

【００６９】

次に、以上のように構成された本実施の形態にかかる複合機１００によるプログラム作成方法について説明する。まず、利用者は、ＰＣ２００でＷＥＢブラウザを起動し、複合機１００上にあるプログラミング画面のＷＥＢページのＵＲＬ（例えば、ｈｔｔｐ：／／ｗｗｗ．ｘｘｘ．ｙｙｙ／ｚｚｚ／ｄｅｂｕｇ．ｈｔｍ）を指定して複合機１００のプログラミングサービス１３２にアクセスする。ＷＥＢブラウザからのアクセスを受けたプログラミングサービス１３２は、プログラミング画面２０１（ｄｅｂｕｇ．ｈｔｍ）をＰＣ２００のＷＥＢブラウザに表示させる。

【００７０】

図７は、プログラミング画面２０１の内容の一例を示す説明図である。図７に示すように、プログラミング画面２０１には、プログラム入力フィールドと、参照ボタンと、アップロードボタンと、保存ボタンと、消去ボタンと、テスト実行ボタンと、実行設定ボタンが表示される。

【００７１】

ここで、プログラミング画面２０１は、ｈｔｍｌ形式（Ｈｙｐｅｒ　Ｔｅｘｔ　Ｍａｒｋｕｐ　Ｌａｎｇｕａｇｅ）で記述されたファイル（ｄｅｂｕｇ．ｈｔｍ）がブラウザによって表示されたものである。また、プログラミングサービス１３２は、プログラミング画面（ｄｅｂｕｇ．ｈｔｍ）からの操作入力や要求に応じたカスタマイズプログラムの入力に関する処理等を、ＣＧＩ（Ｃｏｍｍｏｎ　Ｇａｔｅｗａｙ　Ｉｎｔｅｒｆａｃｅ）のスクリプトを実行することによって行い、その処理結果をプログラミング画面（ｄｅｂｕｇ．ｈｔｍ）に反映して、ＰＣ２００のＷＥＢブラウザに返す。

【００７２】

具体的には、プログラミングサービス１３２は、カスタマイズプログラムの入力処理等の要求（ボタンのクリック操作等）があった場合には、ｄｅｂｕｇ．ｃｇｉのスクリプトを実行する。なお、本実施の形態では、ＣＧＩスクリプトを実行することにより、ＷＥＢアプリ１１１７による各処理を実行しているが、ＣＧ

Ｉスクリプトから呼び出される別のプログラムで実行するように構成しても良い。なお、プログラミング画面を、ＸＭＬ形式（ｅＸｔｅｎｓｉｂｌｅ　Ｍａｒｋｕｐ　Ｌａｎｇｕａｇｅ）で記述するように構成することも可能である。

【００７３】

参照ボタンは、ブラウザが動作しているＰＣ２００の記憶領域に格納されたカスタマイズプログラムを参照するためのボタンであり、この参照ボタンを押下すると、ＰＣ２００の記憶領域内のディレクトリが表示され、所望のディレクトリに存在するカスタマイズプログラムを指定すると、ファイル名フィールドに指定されたカスタマイズプログラムのファイル名が表示される。なお、参照ボタンを押下したときに、複合機１００のＨＤＤ１０３などの記憶媒体のディレクトリを参照するように構成しても良い。

【００７４】

アップロードボタンは、参照ボタンで指定したカスタマイズプログラムあるいは入力フィールドに入力したカスタマイズプログラムを、複合機１００に転送（アップロード）するためにクリック操作するものである。

【００７５】

プログラム入力フィールドは、カスタマイズプログラムを入力する領域である。このプログラム入力フィールドには、ＰＣ２００のキーボードなどの入力装置からカスタマイズプログラムを直接入力する他、上述のように参照ボタンをクリック操作することによってＰＣ２００の所望のディレクトリに存在する既存のプログラムファイルを指定することによってカスタマイズプログラムを入力する。ｄｅｂｕｇ．ｈｔｍにおいて、このプログラム入力フィールドに相当するすべての行の先頭には、「ｐｒｏｇｒａｍ＿ｌｉｓｔ：」の文字列が記述されている。

【００７６】

保存ボタンは、入力したカスタマイズプログラムを複合機１００にアップロードした後に、カスタマイズディレクトリに格納する場合にクリック操作するものである。消去ボタンは、入力フィールドに入力されたカスタマイズプログラムを消去する際にクリック操作するものである。

【００７７】

実行設定ボタンは、カスタマイズディレクトリに保存されたカスタマイズプログラムを複合機１００で実行できるように、カスタマイズプログラムとカスタマイズプログラムを実行させるボタンを対応づける際にクリック操作するものである。この実行設定ボタンが押下されると、プログラミングサービス１３２は、カスタマイズプログラム中で指定されたボタン（キー）を、カスタマイズディレクトリに存在する画面ファイル中のボタン（キー）からキーコードに基づいて検索し、カスタマイズプログラムとボタン（キー）との対応付けを行い、カスタマイズプログラムの名称とボタン（キー）のキーコードのキー対応付けテーブルを生成してＨＤＤ１０３又はフラッシュメモリなどに当該テーブルを格納する。

【００７８】

テスト実行ボタンは、入力されたカスタマイズプログラムを試験的に実行する際にクリック操作するものである。

【００７９】

図８は、入力するカスタマイズプログラムの内容の一例を示す説明図である。図８に示すように、本実施の形態では、カスタマイズプログラムはＵＮＩＸ（登録商標）のシェルスクリプトである。入力されたシェルスクリプトは、汎用ＯＳ１２１（ＵＮＩＸ（登録商標））のコマンド・インタプリタであるシェル１３４によってステップごとに逐次実行される。このため、入力したカスタマイズプログラムのコンパイルおよびリンクは不要である。

【００８０】

また、シェルスクリプトには、開発ライブラリで提供される複合機専用コマンドを指定できるようになっている。図８のスクリプトは、原稿をスキャナで読み込み（ｓｃａｎｉｍａｇｅ）、読み込み時にＴＩＦ形式に圧縮された画像データを伸長し（ｄｅｃｏｍｐ）、伸長した画像データに対し文字認識処理を行う（ｏｃｒ）。そして、文字認識処理による認識結果をテキストデータとしてメール送信（ｍａｉｌ）する例を示している。このスクリプトの中で、ｓｃａｎｉｍａｇｅ，ｄｅｃｏｍｐ，ｏｃｒの各コマンドが開発ライブラリで提供されるコマンドである。また、カスタマイズプログラムには、このシェルスクリプトを実行する

ためのボタン（キー）の指定が可能となっている。

【００８１】

ＰＣ２００のＷＥＢブラウザに表示されているプログラミング画面２０１（ｄｅｂｕｇ．ｈｔｍ）において、参照ボタンをクリックしてＰＣ２００に格納されているカスタマイズプログラムを指定し、あるいはプログラム入力フィールドに直接カスタマイズプログラムを入力し、アップロードボタンをクリック操作した場合、ＷＥＢブラウザから複合機１００のプログラミングサービス１３２に対しリクエストメッセージが送信される。このリクエストメッセージのメッセージボディにはｄｅｂｕｇ．ｈｔｍが含められており、このｄｅｂｕｇ．ｈｔｍには、アップロードボタンをクリック操作したときにｄｅｂｕｇ．ｃｇｉを呼び出す旨が記述されている。このためリクエストメッセージを受信して解析を行ったプログラミングサービス１３２は、ｄｅｂｕｇ．ｃｇｉを実行する。

【００８２】

ここで、リクエストメッセージを受信した複合機１００における処理について説明する。図４は、リクエストメッセージを受信した複合機１００における処理の手順を示すフローチャートである。

【００８３】

ｈｔｔｐｄ１０６は、ポート８０番を常時監視しており、ＰＣ２００からリクエストメッセージをポート８０番で受信する（ステップＳ３０１）。そして、ｈｔｔｐｄ１０６は、受信したリクエストメッセージを読み出し、リクエストメッセージを共有メモリ１０５に書き込む（ステップＳ３０２）。次に、ｈｔｔｐｄ１０６は、リクエストメッセージ受信の旨をＮＣＳ１２８に通知する（ステップＳ３０３）。

【００８４】

通知を受けたＮＣＳ１２８は、さらにリクエストメッセージ受信の旨をプログラミングサービス１３２に通知する（ステップＳ３０４）。プログラミングサービス１３２は、リクエストメッセージ受信の旨の通知を受けたことをトリガとして、共有メモリ１０５を参照して、リクエストメッセージを読み出す（ステップＳ３０５）。そして、リクエストメッセージの内容を解析して（ステップＳ３０

６）、リクエストメッセージのメッセージボディに記述された内容に応じた処理を実行する（ステップＳ３０７）。このとき、メッセージボディのｈｔｍｌ記述にＣＧＩを実行する旨が記述されていれば、対応するＣＧＩを実行する。

【００８５】

そして、プログラミングサービス１３２は、処理の実行結果をレスポンスメッセージとして、ｈｔｔｐｄ１０６経由でリクエストメッセージの送信元であるＰＣ２００ に返信する（ステップＳ３０８）。

【００８６】

次に、プログラミングサービス１３２によるｄｅｂｕｇ．ｃｇｉの処理について説明する。図５は、ｄｅｂｕｇ．ｃｇｉの処理の手順を示すフローチャートである。なお、プログラミングサービス１３２がｄｅｂｕｇ．ｃｇｉのプログラムを含んでいてもよいし、ｄｅｂｕｇ．ｃｇｉがプログラミングサービス１３２の外部に存在してもよい。以下、プログラミングサービス１３２がｄｅｂｕｇ．ｃｇｉのプログラムを含む場合について説明する。

【００８７】

プログラミングサービス１３２は、受信したリクエストメッセージの中のｄｅｂｕｇ．ｈｔｍを先頭から一行づつ読み込む（ステップＳ４０１）。そして、読み込んだ行が「ｐｒｏｇｒａｍ＿ｌｉｓｔ：」の文字列を含むか否かを調べることにより、その行がプログラム入力フィールドか否かを判断する（ステップＳ４０２）。

【００８８】

そして、プログラム入力フィールドであると判断した場合には、プログラムファイルに書き出し（ステップＳ４０３）、読み込んだ一行をそのままＰＣ２００に送信する（ステップＳ４０４）。

【００８９】

そして、読み込んだ行が最終行か否かを判断し（ステップ４０５）、最終行であれば処理を終了し、最終行でなければ上記の処理を次の行に対して行う。これにより、複合機１００において、プログラムファイルが作成され、ＰＣ２００のＷＥＢブラウザには、複合機１００にアップロードされたプログラムの内容が順

次表示されることになる。

【００９０】

上記の方法の他、ＰＣ２００で入力したプログラムを複合機に送信するには種々の方法がある。例えば、アップロードボタンを押すことにより、ＰＣ２００で入力したプログラムをＰＣ２００でプログラムファイルとし、それを複合機に送信する。複合機では、そのプログラムファイルの内容を格納し、必要に応じて格納したプログラムファイルの内容をＰＣ２００に送信する。

【００９１】

このようにプログラミング画面２０１から入力されたカスタマイズプログラムが複合機１００にアップロードされ、プログラム入力フィールドにアップロード後のカスタマイズプログラムが表示された後、保存ボタン、実行設定ボタン、テスト実行ボタンのクリック操作が可能となる。これらのいずれかのボタンをクリック操作した場合におけるプログラミングサービス１３２の処理について説明する。

【００９２】

図６は、プログラミング画面２０１において、保存ボタン、実行設定ボタン、テスト実行ボタンのいずれかのクリック操作が行われた場合のプログラミングサービス１３２の処理の手順を示すフローチャートである。以下の処理は、プログラミングサービス１３２におけるＣＧＩスクリプトとしてのＪａｖａ（登録商標）Ｓｃｒｉｐｔが実行する。すなわち、プログラミング画面２０１におけるボタン操作をＰＣ２００が検出してそのイベントをプログラミングサービスに送信し、プログラミングサービス１３２はそのイベントに応じたＪａｖａ（登録商標）Ｓｃｒｉｐｔを実行する。以下、Ｊａｖａ（登録商標）Ｓｃｒｉｐｔがプログラミングサービス１３２に含まれているものとして説明する。

【００９３】

まず、プログラミングサービス１３２は、クリック操作されたボタンの種類を判断する（ステップＳ５０１）。プログラミング画面２０１で保存ボタンがクリックされた場合には、プログラミングサービス１３２は、ファイル名称を利用者に指定させ、カスタマイズプログラムを指定された名称で複合機１００のＨＤＤ

１０３のカスタマイズディレクトリに格納する（ステップＳ５０２）。

【００９４】

図９は、カスタマイズディレクトリに格納されたカスタマイズプログラムの名称の一例を示す説明図である。図９に示すように、複合機１００のＨＤＤ１０３のカスタマイズディレクトリは、ｈｄｄ／ｘｘｘ／ｏｐｅｐａｎｅとなっており、このカスタマイズディレクトリにカスタマイズプログラムであるｓｈｅｌｌ１，ｓｈｅｌｌ２，ｓｈｅｌｌ３ が格納されている。

【００９５】

一方、プログラミング画面２０１で実行設定ボタンが押下された場合には、プログラミングサービス１３２は、上述のようにカスタマイズプログラムとボタンとの対応付けを行う（ステップＳ５０３）。

【００９６】

図１０は、キー対応付けテーブルの内容の一例を示す説明図である。図１０に示すように、キー対応付けテーブルには、ボタンのキーコードとカスタマイズプログラム名称が対応付けられている。

【００９７】

また、プログラミング画面２０１でテスト実行ボタンが押下された場合には、プログラミングサービス１３２は、格納されたカスタマイズプログラムの実行を行う（ステップＳ５０４）。これは、例えば、プログラミングサービス１３２がカスタマイズプログラをシェルで実行するためのコマンドを実行することによっておこなわれる。

【００９８】

このとき、テスト実行によって、オペレーションパネル２１０に表示される画面と同一の画面をＰＣ２００で動作しているブラウザに表示する。これは、例えば、プログラミングサービス１３２が、プログラムのテスト実行により発生するオペレーションパネル画面への描画命令等を取得し、その描画命令に対応して表示される画面内容と同じ内容をＰＣ２００の画面に表示するような命令をＰＣ２００に対して送信することにより実現できる。

【００９９】

テスト実行によってエラーが出た場合には、ＰＣ２００でプログラムの修正を行い、修正したプログラムをアップロードし、再びテスト実行をする。

【０１００】

次に、このようにして生成されたカスタマイズプログラムの実行処理について説明する。図１１は、複合機１００の電源投入が行われてから、カスタマイズプログラムの実行が可能となるまでの一連の処理の手順を示すフローチャートである。

【０１０１】

カスタマイズプログラムのカスタマイズディレクトリへの保存および起動設定ファイル２１１への設定が完了したら、利用者は複合機１００の電源を再投入する。このとき複合機１００では、ＲＯＭモニタ（図示せず）によってハードウェアの診断処理を行い（ステップＳ１００１）、その後汎用ＯＳ１２１を起動し（ステップＳ１００２）、さらに複合機初期化部１２９を起動する（ステップＳ１００３）。複合機初期化部１２９は、まずコントロールサービスの起動を行い（ステップＳ１００４）、続いてコピーアプリ１１２、プリンタアプリ１１１などの複合機１００の出荷時に提供されている既存アプリを起動し（ステップＳ１００５）、さらに出荷後新たに開発された外部アプリなどのプログラムを起動するためにプログラム起動部１３１を起動する（ステップＳ１００６）。

【０１０２】

プログラム起動部１３１は、まずＨＤＤ１０３の起動設定ファイル２１１を参照し（ステップＳ１００７）、起動すべきプログラムの名称を取得する。

【０１０３】

図１２は、起動設定ファイル２１１の一例を示す説明図である。図１２に示すように、起動設定ファイル２１１には、「プログラム名称　カスタマイズプログラム名」の形式で、プログラム起動部１３１によって起動されるプログラムが登録されている。図１２の例では、シェルスクリプトであるシェルのプログラムが最初に登録され、その後に、外部アプリとしてｘｘｘアプリが登録されている。

【０１０４】

このため、図１２の例では、プログラム起動部１３１は、まずシェルを起動し

、次いで、外部アプリであるＸＸＸアプリを順に起動するように設定されている。このため、プログラム起動部１３１は、まずインタプリタ１３４（シェル）を起動し（ステップＳ１００８）、次いで、外部アプリであるＸＸＸアプリを順に起動する（ステップＳ１００９）。

【０１０５】

オペレーションパネル２１０の操作表示部に表示されたボタン（キー）の中で、カスタマイズプログラムが割り付けられたボタン（キー）を押下（タッチ操作）すると、シェルスクリプトで記述されたカスタマイズプログラムがシェルによって実行される。これは例えば次のようにして実行される。

【０１０６】

すなわち、カスタマイズプログラムが割り付けられたボタン（キー）の押下情報をプログラミングサービス１３２が取得し、キー対応付けテーブルを参照することによりそのボタンに対応するカスタマイズプログラムファイル名を取得する。そして、そのカスタマイズプログラムをシェルで実行するための命令を実行する。なお、上記の起動設定ファイルにカスタマイズプログラムを設定することにより、カスタマイズプログラムを実行してもよい。

【０１０７】

このように実施の形態１にかかる複合機１００では、プログラミングサービス１３２によって、ネットワーク上のＰＣ２００に、カスタマイズプログラムのプログラミング画面を表示し、この画面からカスタマイズプログラムを入力させ、複合機１００のシェル１３４によって、カスタマイズプログラムを実行する。これにより、カスタマイズプログラムの開発において、プログラミング画面から追加、修正ができ、また追加修正による再コンパイル、再リンクなしに複合機１００で実行でき、プログラムの作成を容易に行うことができる。

【０１０８】

また、実施の形態１にかかる複合機１００では、プログラミングサービス１３２によって、インターネットに接続されたＰＣ２００に対しプログラミング画面を表示し、この画面からカスタマイズプログラムを入力させているので、カスタマイズプログラムの入力を入力が容易なＰＣ２００で行うことができ、複合機１

００上でカスタマイズプログラムの作成編集を行う場合に比べて、カスタマイズプログラムの作成をより効率的に行うことができる。

【０１０９】

なお、実施の形態１にかかる複合機１００では、インターネット２２０に接続されたクライアント端末としてのＰＣ２００の要求により、カスタマイズプログラムの入力処理を行っているが、インターネット以外のネットワーク、例えばＬＡＮなどに接続されたクライアント端末からカスタマイズプログラムの入力を行うように構成することも可能である。また、通信プロトコルとしては、ＴＣＰ／ＩＰ以外のプロトコルを利用しても良い。

【０１１０】

また、上記の実施の形態では、複合機からＰＣ２００に対してプログラム入力画面を送信しているが、ＰＣ２００が本実施の形態におけるプログラムの入力、送信などを行うプログラムを備えていてもよい。

【０１１１】

（実施の形態２）

実施の形態１にかかる複合機１００は、ネットワーク接続されたクライアント端末であるＰＣ２００でカスタマイズプログラムを作成して複合機１００に保存するものであったが、この実施の形態２にかかる複合機は、カスタマイズプログラムの入力を複合機１００のオペレーションパネルから行うものである。

【０１１２】

図１３は、実施の形態２にかかる複合機１１００における主要構成を示すブロック図である。なお、複合機１１００の他の構成については、図２に示した実施の形態１の複合機１００の構成と同様である。図１３に示すように、本実施の形態にかかる複合機１１００は、複合機初期化部１２９と、プログラム起動部１３１と、プログラミングサービス１１３２と、インタプリタ（シェル）１３４と、ＨＤＤ１０３とを備えた構成となっている。

【０１１３】

プログラミングサービス１１３２は、オペレーションパネル２１０の操作表示部にプログラミング画面を表示し、表示されたプログラミング画面からカスタマ

イズプログラムを入力させるものである。また、プログラミングサービス１１３２は、入力されたカスタマイズプログラムをＨＤＤ１０３のカスタマイズ領域として作成されたカスタマイズディレクトリ２１２に保存する。さらに、プログラミングサービス１１３２は、複合機１１００上でカスタマイズプログラムが実行されるように、起動設定ファイルに入力されたカスタマイズプログラムを登録することもできる。

【０１１４】

なお、複合機初期化部１２９、プログラム起動部１３１、シェル１３４の機能およびＨＤＤ１０３の起動設定ファイル２１１およびカスタマイズ領域としてのカスタマイズディレクトリについては、実施の形態１にかかる複合機１００と同様である。

【０１１５】

図１４は、プログラミング画面１１０１の一例を示す説明図である。本実施の形態にかかる複合機１１００において、カスタマイズプログラムを作成するためには、例えば、システム初期設定画面からプログラミングのタグを選択してプログラミング画面を表示させる。

【０１１６】

このプログラミング画面１１０１は、実施の形態１における同画面とほぼ同様に、入力フィールドと、参照ボタンと、保存ボタンと、実行設定ボタンと、テスト実行ボタンとが表示されている。かかるプログラミング画面１１０１では、タッチ操作でキー入力を行うソフトウェアキーボードを操作表示部に表示させて、入力フィールドにカスタマイズプログラムを入力することできる。

【０１１７】

しかしながら、操作表示部の領域サイズや入力の困難さから直接カスタマイズプログラムを入力することは困難である。このため、予めＨＤＤ１０３などの記憶媒体の所望の領域に、ＰＣなどで作成したカスタマイズプログラムをネットワーク経由で格納し、プログラミング画面１１０１上の参照ボタンを押下して、カスタマイズプログラムを選択することにより、入力フィールドにカスタマイズプログラムを表示し、ソフトウェアキーボードを表示させ、プログラムの簡単な修

正などだけを入力フィールドで行うことが好ましい。

【０１１８】

そして、カスタマイズプログラムが完成したら、保存ボタンを押下する。これによって、実施の形態１と同様に、プログラミングサービス１１３２によって、カスタマイズプログラムをＨＤＤ１０３のカスタマイズディレクトリに保存する。また、実行設定ボタンを押下すると、実施の形態１と同様に、プログラミングサービス１１３２によって、カスタマイズプログラムとカスタマイズプログラムで指定されたボタン（キー）のキーコードが対応付けられてテーブルとして保存される。なお、作成したカスタマイズプログラムの実行処理については、実施の形態１の複合機１００ と同様である。

【０１１９】

このように実施の形態２にかかる複合機１１００では、プログラミングサービス１１３２によって、オペレーションパネル２１０の操作表示部にプログラミング画面１１０１を表示し、この画面からカスタマイズプログラムを入力させているので、ネットワークに接続されていないスタンドアローンの複合機１１００においてもカスタマイズプログラムを作成することができる。

（実施の形態３）

実施の形態１および２の複合機１００，１１００は、カスタマイズプログラムと操作表示部のボタン（キー）との対応付けをカスタマイズプログラムの記述により行うものであったが、この実施の形態３にかかる複合機は、複合機の電源投入時にカスタマイズプログラムと操作表示部のボタン（キー）との対応付けを動的に行うものである。

【０１２０】

図１５は、実施の形態３にかかる複合機１３００の機能的構成を示すブロック図である。本実施の形態の複合機１３００では、実施の形態１と同様に、ネットワーク上のＰＣ２００でカスタマイズプログラムの作成を行って、複合機１３００のＨＤ１０３に作成されたカスタマイズプログラムを保存して実行する。

【０１２１】

図１５に示すように、本実施の形態の複合機１３００は、オペパネ登録部１３

３を備えている点が図２に示す実施の形態１の複合機１００と異なっており、他の構成は実施の形態１の複合機１００と同様である。

【０１２２】

オペパネ登録部１３３は、カスタマイズプログラムを実行させるためのボタン（キー）の割り付けを行うものである。すなわち、オペパネ登録部１３３によって割り付けたボタン（キー）をオペレーションパネル上で押下（タッチ操作）することによって、カスタマイズプログラムが実行される。

【０１２３】

ＰＣ２００上からのカスタマイズプログラムの作成および保存の処理は、実施の形態１のプログラミングサービス１３１と同様に行われる。

【０１２４】

以下、カスタマイズプログラムと操作表示部に表示されるボタン（キー）の割付処理について説明する。カスタマイズプログラムのカスタマイズディレクトリへの保存が完了すると、作成したカスタマイズプログラムをオペレーションパネル２１０のボタン（キー）に割り付けるために、利用者は複合機１００の電源を再投入する。

【０１２５】

図１６は、複合機１３００の電源投入が行われてから、カスタマイズプログラムとボタン（キー）との割り付けが完了するまでの一連の処理の手順を示すフローチャートである。

【０１２６】

複合機１３００の電源投入からプログラム起動部１３１までの処理（ステップＳ１５０１～Ｓ１５０６）については、実施の形態１の複合機１００による処理（ステップＳ１００１～Ｓ１００６）と同様である。

【０１２７】

起動されたプログラム起動部１３１は、まずＨＤＤ１０３の起動設定ファイル２１１を参照し（ステップＳ１５０７）、起動すべきプログラムの名称を取得する。図１７は、本実施の形態で使用される起動設定ファイル２１１の内容の一例を示す説明図である。この例では、オペパネ登録部のプログラム、シェルおよび

外部アプリであるＸＸＸ アプリを順に起動するように設定されている。

【０１２８】

このため、プログラム起動部１３１は、まずオペパネ登録部１３３のプログラムを起動する（ステップＳ１５０８）。なお、このオペパネ登録部１３３のプログラムは、予めカスタマイズディレクトリに格納されている。次いで、インタプリタ（シェル）１３４を起動し（ステップＳ１５０９）、最後に、登録されている外部アプリを起動する（ステップＳ１５１０）。

【０１２９】

起動されたオペパネ登録部１３３は、キー割り付け設定画面をオペレーションパネルに表示する（ステップＳ１５１１）。図１８は、キー割り付け設定画面の一例を示す説明図である。図１８に示すように、キー割り付け設定画面には、各カスタマイズプログラムごとに割り付け可能なボタン（キー）が表示され、かかる画面で所望のボタン（キー）を押下（タッチ操作）すると、オペパネ登録部１３３は、該当するカスタマイズプログラムの名称と押下されたボタン（キー）のキーコードとを対応付けて、ＨＤＤ１０３又はフラッシュメモリなどの記憶媒体にテーブル情報として格納する（ステップＳ１５１２）。これによって、オペレーションパネルに表示されるボタン（キー）にカスタマイズプログラムが割り付けられることになる。

【０１３０】

このように実施の形態３にかかる複合機１３００では、プログラミングサービス１３２によって、プログラミング画面２０１を表示してカスタマイズプログラムを入力させ、オペパネ登録部１３３によって、カスタマイズプログラムを操作表示部に表示および入力を行うボタン（キー）に割り付ける。このオペパネ登録部１３３を複合機１３００の電源投入時に起動することにより、カスタマイズプログラムが実行される場合のボタン操作を複合機１３００の起動時に動的に定めることができる。

【０１３１】

なお、本実施の形態では、オペパネ登録部１３３によってボタン（キー）への割り付け処理を行っているが、例えば、ＳＣＳ１２２などのコントロールサービ

スでボタン（キー）割り付け処理を行うようにしてもよい。これにより、割り付け処理のために別途プログラムを起動する必要が無くなり、効率的な処理が行える。

【０１３２】

（実施の形態４）

実施の形態１～３の複合機１００，１１００，１３００は、シェルスクリプトとしてカスタマイズプログラムを複合機１００，１３００の汎用ＯＳ１２１で用意されたコマンドインタプリタであるシェル１３４によって実行するものであったが、この実施の形態４にかかる複合機は、シェル１３４以外のインタプリタを使用してカスタマイズプログラムを実行するものである。

【０１３３】

図１９は、実施の形態４にかかる複合機１７００を備えたネットワーク構成図であり、図２０は、実施の形態４にかかる複合機１７００の機能的構成を示すブロック図である。実施の形態４にかかる複合機１７００は、実施の形態１と同様に、ＴＣＰ／ＩＰを利用したインターネットに接続され、当該ネットワークに接続されたクライアント端末であるＰＣ２００からカスタマイズプログラムの入力およびデバッグ作業を行うことを可能としたものである。なお、カスタマイズプログラムとしては、例えば、オペレーションパネル２１０の操作表示部にボタンを表示し、あるボタンを押したときにそのボタンに対応した処理を行うためのプログラムがある。

【０１３４】

本実施の形態の複合機１７００では、カスタマイズプログラムを、Ｖｉｓｕａｌ Ｂａｓｉｃ（登録商標）言語で記述する。そして、作成されたカスタマイズプログラムを、汎用ＯＳ１２１の配下で動作するＶｉｓｕａｌ Ｂａｓｉｃインタプリタで実行させている。

【０１３５】

図１９に示すように、複合機１７００は、複合機初期化部１２９と、プログラム起動部１３１と、プログラミングサービス１３２と、Ｖｉｓｕａｌ Ｂａｓｉｃインタプリタ１７３４（以下、「ＶＢインタプリタ１７３４」という。）と、

ＮＣＳ１２８と、ｈｔｔｐｄ１０６と、共有メモリ１０５と、ＨＤＤ１０３とを備えた構成となっている。なお、プログラミングサービス１３２、複合機初期化部１２９、プログラム起動部１３１、ＮＣＳ１２８、ｈｔｔｐｄ１０６の構成および機能、ＨＤＤ１０３に格納される起動設定ファイル２１１の内容およびカスタマイズディレクトリについては、実施の形態１の複合機１００と同様である。

【０１３６】

ＶＢインタプリタ１７３４は、カスタマイズプログラムをステップごとに解析し逐次実行するものである。本実施の形態において、カスタマイズプログラムの作成および保存は、実施の形態１のプログラミングサービス１３２と同様に行われる。

【０１３７】

また、本実施の形態の複合機１７００では、ＶＢインタプリタ１７３４によってカスタマイズプログラムを実行させるため、複合機１７００の起動時にＶＢインタプリタ１７３４も起動させておく必要がある。このため、実施の形態１と同様の起動設定ファイル２１１にＶＢインタプリタ１７３４のプログラム名称を予め設定しておく。

【０１３８】

図２１は、起動設定ファイルの内容の一例を示す説明図である。図２１に示すように、本実施の形態の複合機１７００で使用する起動設定ファイル２１１には、先頭の行に「ＶＢインタプリタプログラム」が設定されている。このため、プログラム起動部１３１は、起動設定ファイル２１１を参照した後、まずＶＢインタプリタ１７３４を起動することになり、これによってカスタマイズプログラムの実行が可能となる。

【０１３９】

このように実施の形態４にかかる複合機１７００では、汎用ＯＳ１２１の配下で動作するＶＢインタプリタによってカスタマイズプログラムを実行するので、汎用ＯＳで提供されるインタプリタよりも高度な処理を行えるインタプリタを利用することができる。従って、ＵＮＩＸ（登録商標）のシェルを用いるよりも高度なプログラムを容易に作成することが可能となる。

【０１４０】

なお、実施の形態４では、ＰＣ２００上ではＶＢインタプリタはインストールされていないが、ＰＣ２００にＶｉｓｕａｌ　Ｂａｓｉｃ開発環境をインストールし、操作表示部の画面設計を含むプログラム作成をＰＣ２００上で行って、生成されたプログラムをファイル転送などを利用して、複合機１７００のカスタマイズディレクトリに格納するように構成しても良い。

【０１４１】

また、実施の形態４の複合機１７００では、インタプリタとしてＶＢインタプリタ１７３４を使用し、開発するカスタマイズプログラムをＶｉｓｕａｌ　Ｂａｓｉｃ言語で記述していたが、この他、インタプリタ用の言語を用いても良い。例えば、複合機１７００にＪａｖａ（登録商標）ＶＭ（仮想マシン）を搭載し、カスタマイズプログラムをＪａｖａ（登録商標）言語で記述するように構成することもできる。

【０１４２】

実施の形態１～４にかかる複合機では、プログラミングサービス１３２をコントロールサービス層に搭載した構成としているが、アプリケーション層に搭載した構成とすることもできる。

【０１４３】

（実施の形態５）

実施の形態４にかかる複合機は、汎用ＯＳ１２１の配下で動作するインタプリタを使用してカスタマイズプログラムを実行するものであったが、この実施の形態５にかかる複合機１８００は、ＶＢインタプリタ１８０１とＶＢアプリ１８０２をアプリケーション層に搭載した構成としたものである。

【０１４４】

図２２に、実施の形態５に係る複合機の機能構成のブロック図を示す。実施の形態５にかかる複合機は、実施の形態４と同様に、ＴＣＰ／ＩＰを利用したインターネットに接続され、当該ネットワークに接続されたクライアント端末であるＰＣ２００から、カスタマイズプログラムの入力およびデバッグ作業を行うことを可能としたものである。ただし、実施の形態５では、作成されたカスタマイズ

プログラム（以下、ＶＢアプリと称する）を、アプリケーション層に搭載されたＶｉｓｕａｌ　Ｂａｓｉｃインタプリタ（以下、ＶＢインタプリタと称する）で実行させている。なお、図中、プログラミングサービス１３２をアプリケーション層に搭載した構成としてもよい。

【０１４５】

Ｖｉｓｕａｌ　Ｂａｓｉｃ開発環境をインストールしたＰＣ２００上でＶＢアプリの作成を行い、その作成したＶＢアプリを複合機に転送し、複合機でそのＶＢアプリを格納しておき、オペレーションパネルにおけるアプリ選択等に応じてＶＢアプリを起動してＶＢインタプリタ１８０１により実行するようにする。

【０１４６】

以下、より具体的に説明する。

【０１４７】

図２３は、ＰＣ２００上でＶｉｓｕａｌ　Ｂａｓｉｃ開発環境を用いてＶＢアプリを作成している場面の例を示す図である。

【０１４８】

同図中の画面内にはProject２-Form２(Form)の表題がついたウィンドウと、Project２-Form２(コード)の表題がついたウィンドウが示されている。

【０１４９】

Project２-Form２(Form)の表題がついたウィンドウは、複合機のオペレーションパネルで表示されるメニュー画面を作成するためのものであり、表示されている状態では、３つのボタンが設定されており、一つは"SCAN"、一つは"FAX"あと一つは"START"のボタンである。これらのボタンが押された場合に、複合機のスキャニングやファックス送信などの動作が行われるように設計される。

【０１５０】

Project２-Form２(コード)の表題がついたウィンドウは、上記のパネルの画面中にあるボタンが押されたときの動作を記述するプログラミングを行うためのものである。

【０１５１】

図２３に示す状態でプログラムの保存を行うと、Project２.vbpとForm２.frm

という二つのファイルが生成される。Project２.vbpにはプロジェクトを管理するための情報や、プログラムを実行する際の条件となるものが、変数への値の設定として保存されている。

【０１５２】

Form２.frmは、ＶＢインタプリタ１８０１によりインタプリトされるプログラムそのものであり、このＶＢアプリをＰＣ２００から複合機上のプログラミングサービス１３２を介してアップロードする。

【０１５３】

そして、プログラミング画面２０１でテスト実行ボタンが押下された場合には、複合機上のＶＢインタプリタ１８０１が起動され、入力されたＶＢアプリの実行が行われる。このとき、テスト実行によって、オペレーションパネル２１０に表示される画面と同一の画面をＰＣ２００で動作しているブラウザに表示する。

【０１５４】

図２４、２５に、ＶＢインタプリタ１８０１により実行されるＶＢアプリの例を示す。

【０１５５】

図２４に示すＶＢプログラムが複合機において実行されると、まず、ＮＥＸＴＷＩＮボタンを含むウィンドウが表示される。そして、ＮＥＸＴＷＩＮボタンが押されると、call("change_display", "Form2")により、関数change_display("Form2")が実行され、次の画面（Form２ウィンドウ）に切り替わる。

【０１５６】

図２５に、そのForm２ウィンドウを表示するＶＢプログラムを示す。なお、Form２のＶＢプログラムは、図２３に示した画面に対応する。

【０１５７】

Form２ウィンドウでは、ＳＣＡＮボタンが押されたときに、

```
call("ScanStart", A4, ADF, TIFF, BINARY, "/work/tmpfile.tif")
gwOpItemCreate(win, MESSAGE_ITEM,
                    ITEM_MESSAGE_X,        0,
                    ITEM_MESSAGE_Y,        12,
```

```
ITEM_MESSAGE_WIDTH,     100,
ITEM_MESSAGE_HEIGHT,    12,
ITEM_MESSAGE_CENTER_X,  0,
ITEM_MESSAGE_BLINK,     0,
ITEM_MESSAGE,           "文字列", 0,
ITEM_MESSAGE_FONT,      FONT_12, 0,
0);
```

が実行される。

【０１５８】

最初のScanStart関数は、A4サイズの原稿をADFから読み込んで、２値画像としてTIFFフォーマットでHDD上のファイル"/work/tempfile"で保管することを意味している。この関数をＶＢインタプリタ１８０１が解釈して、複合機に動作を行わせることとなる。

次の行のgwOpItemCreate()は複合機におけるＯＣＳ１２６によりその機能が提供されている関数であり、現在表示されている画面に文字列を表示する機能を実行するものである。gwOpItemCreate()は、ＶＢインタプリタ１８０１により、ＯＣＳ１２６への関数コールとして実行される。

【０１５９】

このようにして作成されたＶＢアプリを複合機で実行させるには、図１１に示した起動設定ファイルにＶＢインタプリタとＶＢアプリを設定し、例えば図１０に示した手順と同様にして、ＶＢインタプリタとＶＢアプリを起動する。

【０１６０】

複合機でのＶＢアプリ実行時、オペレーションパネルに表示操作を行うためには次のような処理を行う。まず、ＶＢインタプリタ１８０１は、ＳＣＳ１２２からの画面描画準備完了イベントを待ち、上記のイベントを受け取ったら、初期表示するウィンドウイメージを作成する（ルートウィンドウ生成）。次に、ＶＢアプリが、ルートウィンドウ内の子ウィンドウの生成や文字列表示の関数を実行することにより初期画面を準備する。そして、画面準備完了を示す関数を実行することにより、画面の準備ができたことをＳＣＳ１２２に通知する。次に、オペレ

ーションパネル上でアプリ選択ボタンが選択されると、ＳＣＳ１２２から操作部オーナー移行要求のイベントが発行される。このイベントをＶＢアプリが受け取ると、操作部オーナー移行可能であることを示す関数を実行し、この時点で上記の初期画面が表示される。この後、ＶＢアプリの実質的な処理がＶＢインタプリタ１８０１により実行される。

【０１６１】

上記のように、本実施の形態によれば、ＶＢインタプリタを用いることにより効率良くアプリを開発することが可能となる。

（実施の形態６）

実施の形態５にかかる複合機は、ＶＢインタプリタをアプリケーション層に搭載した構成としたものであったが、実施の形態６にかかる複合機は、Ｊａｖａ（登録商標）実行環境をアプリケーション層に搭載した構成としたものである。

【０１６２】

図２６に、実施の形態６に係る複合機１９００の機能構成のブロック図を示す。同図に示すように、実施の形態６に係る複合機１９００は、アプリケーション層にＪａｖａ（登録商標）実行環境１９０１とＪａｖａ（登録商標）アプリ１９０２を搭載した構成を有している。Ｊａｖａ（登録商標）アプリケーションをＰＣ（パーソナルコンピュータ）などで開発し、それを複合機にダウンロードしてＪａｖａ（登録商標）実行環境１９０１を用いて実行する。

【０１６３】

図２７は、ＰＣにおけるＪａｖａ（登録商標）開発環境と複合機１９００におけるＪａｖａ（登録商標）実行環境の構成を示す図である。

【０１６４】

同図に示すように、複合機１９００におけるＪａｖａ（登録商標）実行環境は、クラスライブラリ１９１１、仮想マシン１９１２、アプリケーション管理部１９１４を有している。

【０１６５】

クラスライブラリ１９１１は、Ｊａｖａ（登録商標）アプリが容易に複合機を操作するためのサービスを提供するために用いられるクラスライブラリである。

本実施の形態におけるクラスライブラリは、例えば、操作パネルクラス、イベントクラス、複合機制御クラス、ネットワークトランザクションクラス、基本的なＪａｖａ（登録商標）クラスライブラリ等から構成される。なお、Ｊａｖａ（登録商標）アプリ自身もクラスの集合であり、そのクラスがクラスライブラリのクラスを呼び出すことにより、アプリの処理が実行される。Ｊａｖａ（登録商標）アプリに、それが使用するクラスライブラリのクラスの機能を全て含めることにより、クラスライブラリ１９１１を搭載しない構成もあり得る。

【０１６６】

Ｊａｖａ（登録商標）のソースコードプログラムはコンパイルして中間コード形式のバイトコードされ、複合機にロードされる。仮想マシン１９１２はそのバイトコードを解釈・実行する。

【０１６７】

アプリケーション管理部１９１４は、Ｊａｖａ（登録商標）アプリを管理する機能を有しており、例えば、Ｊａｖａ（登録商標）アプリのリスト表示、起動や強制終了などのＪａｖａ（登録商標）アプリ実行管理、Ｊａｖａ（登録商標）アプリのロードやバージョンアップ、インストール済みのＪａｖａ（登録商標）アプリの削除、アプリケーション登録のためのパスワード設定等を行う機能を有している。

【０１６８】

より詳細には、図２８に示すように、Ｊａｖａ（登録商標）の実行環境は更にネイティブプログラムインタフェース１９１３を有している。ネイティブプログラムインタフェース１９１３は、仮想マシン１９１２で実行されるＪａｖａ（登録商標）コードが、Ｃ言語などの他のプログラミング言語で書かれたアプリケーションやライブラリと相互運用するための機能を有している。このネイティブプログラムインタフェース１９１３の機構を用いて、アプリはコントロールサービスのＡＰＩにアクセスできる。

【０１６９】

図２７に示すＪａｖａ（登録商標）開発環境は、Ｊａｖａ（登録商標）の実行環境と同様のクラスライブラリ１９２１、仮想マシン１９２２、アプリケーショ

ン管理部１９２４を有している。また、更に、バイトコードを生成するためのＪａｖａ（登録商標）コンパイラ１９２５と、アプリによる複合機の操作をエミュレートするエミュレータ１９２３を有している。このＪａｖａ（登録商標）開発環境で開発されたＪａｖａ（登録商標）アプリは、ＩＣカード等の記録媒体を介して複合機のＪａｖａ（登録商標）実行環境にロードできる。また、ネットワークを介してロードすることもできる。

【０１７０】

例えば、図２９に示すように、ＷｅｂサーバにＪａｖａ（登録商標）開発環境で開発したＪａｖａ（登録商標）アプリを格納しておき、Ｊａｖａ（登録商標）実行環境からＷｅｂサーバにアクセスしてＪａｖａ（登録商標）アプリをロードするように構成できる。また、Ｗｅｂサーバの代わりにＦＴＰサーバを使用することもできる。

【０１７１】

（Ｊａｖａ（登録商標）アプリケーションの開発）

次に、Ｊａｖａ（登録商標）開発環境でのＪａｖａ（登録商標）アプリの開発について詳細に説明する。

【０１７２】

図３０は、Ｊａｖａ（登録商標）アプリ開発の流れを示す図である。このように、まず、Ｊａｖａ（登録商標）のソースコードをＪａｖａ（登録商標）ファイルとして作成し、これをＪａｖａ（登録商標）コンパイラでコンパイルし、クラスファイルとする。そして、複数のクラスをまとめてＪａｒファイルとする。なお、Ｊａｒファイルにはｍａｉｎルーチンを持つクラスを示すファイルを含めておくことが好ましい。

【０１７３】

このようにして作成したＪａｖａ（登録商標）アプリを、これまでに説明した実施の形態と同様にして複合機上でテスト実行することにより、デバッグを行うことが可能であるが、本実施の形態では、図２７に示すエミュレータ１９２３を用いることにより、ターゲットである複合機を用いずにＪａｖａ（登録商標）アプリのデバッグをすることが可能である。これは、Ｊａｖａ（登録商標）では、

仮想マシンを使用する構成となっているため、実行環境をエミュレートすることが比較的容易であることによる。エミュレータを用いた開発が完了したら、Ｊａｖａ（登録商標）アプリ（バイトコード）をネットワーク又はＩＣカードを介して複合機にロードすることにより、複合機を用いた評価を行うことが可能である。

【０１７４】

エミュレータは、Ｊａｖａ（登録商標）アプリ側からの命令を受信し、それに応じた動作を行うように記述したプログラムとして構成することができる。例えば、複合機のオペレーションパネルの表示部への表示命令に対しては、複合機上に表示する画面と同様の画面をＰＣのディスプレイ上に表示する。ただし、開発環境であるＰＣには複合機におけるエンジン（スキャナ、プリンタなど）がないので、例えば、Ｊａｖａ（登録商標）アプリからのプリント命令に対しては、ＰＣの画面上にプリントを行う画像を表示したり、ＰＣ自身に接続されているデフォルトのプリンタに印刷を行う。また、ユーザがコピー操作をＰＣのエミュレータ画面上でした場合には、例えば、ユーザにコピー対象となるファイルをローカルファイルシステムから選ぶように通知し、コピーのあて先を尋ねるダイアログボックスを表示し、選択されたファイルをその指定された宛先にコピーする。

【０１７５】

なお、エミュレータは、図２７に示すように仮想マシンとクラスライブラリとは別のプラグラムとして実装することもできるし、図３１に示すように、クラスライブラリの中のクラスとして実装することもできる。

【０１７６】

図３２に、エミュレータによりＰＣに表示されるオペレーションパネル画面を示す。この画面に表示されるボタンなどの要素は、実際の複合機のものと同等である。また、エミュレータは、格納したＪａｖａ（登録商標）アプリを選択して実行するローダーの機能も有している。ローダーとして動作する場合には、例えば、図３３に示す画面を表示する。そして、ユーザはＪａｖａ（登録商標）アプリを選択し、実行ボタンを押すことによりＪａｖａ（登録商標）アプリが実行される。なお、ここで表示されるＪａｖａ（登録商標）アプリは、ｍａｉｎルーチ

ンを持つクラス名、もしくはＪａｒファイル名である。

【０１７７】

エミュレータは、エミュレータ上で動作するアプリケーションの状態状態情報を自動的にログファイルに収集する機能も有しており、どのようなメッセージをログとして残すかの設定もできる。例えば、アプリを異常終了させるような深刻なエラーのみをログとして収集するように設定したり、アプリケーションのデバッグに有益な詳細情報をログとして収集するように設定することができる。

【０１７８】

エミュレータを用いて開発環境側でプログラムのデバッグを行うことは、他の実施の形態においても適用可能である。すなわち、例えば実施の形態１において、ＰＣ２００にインタプリタと、本実施の形態と同様のエミュレータを搭載する。プログラムのデバッグが進んだところで、評価のために複合機にアクセスしてプログラムを送信する。

【０１７９】

（Ｊａｖａ（登録商標）アプリケーションの例）

次に、Ｊａｖａ（登録商標）アプリケーションの例について説明する。ここでは、操作パネルクラスを使用したサンプルプログラムを使用したＪａｖａ（登録商標）アプリの例について説明する。図３４に、操作パネルクラスの階層構成を示す。

【０１８０】

図３４中、＊＊＊Ｗｉｎｄｏｗのクラスはウィンドウ作成のためのクラスであり、＊＊＊＊Ｉｔｅｍのクラスは、ボタンなどのウィンドウ中におけるエレメントを作成するためのクラスである。例えば、以下に示すコードを含むプログラムにより、図３５に示すようなメインウィンドウとその中のエレメントが作成される。

【０１８１】

```
mainwindow = new PanelWindow (panel.root(), 640, 240);   (a)
MessageItem msg = new MessageItem (20, 20, 600, 40);     (b)
ButtonItem Button = new ButtonItem (10, 60, 80, 30);     (c)
```

上記の（ａ）は、幅６４０ドット、高さ２４０ドットのメインウィンドウを作成するためのコードである。（ｂ）は、メインウィンドウの位置（２０、２０）に幅６００、高さ４０のメッセージアイテムを作成するためのコードである。（ｃ）は、（１０、６０）の位置に、サイズ８０×３０のボタンを作成するためのコードである。

【０１８２】

図３６に示すプログラムは、複合機のオペレーションパネルにユーザーインタフェースを実装するプログラムである。図３７に示すように、このプログラムは起動直後に”Hello World”及び”Your Input Was…”のメッセージと”Get Input…”のボタンを表示する。ユーザーが”Get Input…”のボタンに触れると、パネル上にソフトキーボードを表示し、ソフトキーボードには、”Add your Input”のタイトルが表示され、ユーザーに任意のコード入力を促す。ユーザーが、ソフトキーボードから任意の文字列、例えば、“ABCD!”を入力した場合、パネル上の”Your Input Was…”のメッセージの下にユーザーが入力した文字列を表示、すなわち、“ABCD!”を表示する。

【０１８３】

以下、図３６に示すサンプルプログラムの内容について、プログラム中に記述した注釈の番号に沿って説明する。

【０１８４】

①は、操作パネルクラスライブラリを使用するための命令である。②は、GWAppを継承することを示し、これにより複合機のアプリケーション雛型（抽象）を使用することが可能となる。また、複雑な初期設定や終了処理をユーザーが記述せずに済み、メッセージ受信などの処理もユーザーに対して隠蔽することができる。また、③ により“Hello World”のメッセージを表示するオブジェクトを生成し、④ により“Get Input...”のボタンオブジェクトを生成する。そして、⑤ により“Your Input Was...” のメッセージを表示するオブジェクトを生成し、⑥ によりソフトキーボードから入力した文字を表示するメッセージオブジェクトを生成する。

【０１８５】

（Ｊａｖａ（登録商標）実行環境におけるＪａｖａ（登録商標）アプリの実行について）

図２７に示す構成において、アプリケーション管理部１９１４の機能を用いることにより、以下のような手順でダウンロードから起動までの処理を行うことが可能である。

【０１８６】

アプリケーション管理部１９１４は、他のアプリケーションと同様にして複合機の起動時に起動され、アプリケーション起動キー等を押すことにより、オペレーションパネルにユーザーインタフェースを表示する。このとき、Ｊａｖａ（登録商標）アプリがロードされていない場合は、アプリケーションロード画面を表示する。ユーザーは、このアプリケーションロード画面から所定のＷｅｂサイト又は、ＩＣカードにアクセスし、Ｊａｖａ（登録商標）アプリをロードする。

【０１８７】

ロードの処理においては、Ｊａｖａ（登録商標）アプリをロードする前に、当該Ｗｅｂサーバ等に対してＪａｖａ（登録商標）アプリに関する情報の問い合わせを行い、インストール可能であるＪａｖａ（登録商標）アプリであるか否かを確認する。確認項目は、例えば、プログラムサイズ、バージョン確認、プログラム最終更新情報、使用するメモリワークサイズ、使用するストレージサイズ、あらかじめ使用するネットワークアドレスの確認、利用可能期間（利用回数）、アプリケーションプログラム名等である。

【０１８８】

上記の事項を確認し、インストール可能であると判断した場合は、アプリケーション管理部１９１４がＪａｖａ（登録商標）アプリケーションファイルを複合機にダウンロードする。

【０１８９】

そして、Ｊａｖａ（登録商標）アプリケーションプログラムのダウンロードが終了すると、アプリケーション管理部１９１４は、アプリケーション名を取得し、アプリケーションリストに追加し、同時にＪａｖａ（登録商標）アプリケーションプログラムを複合機のハードディスクに格納する。

【０１９０】

以上の処理が完了することにより、ダウンロードしたＪａｖａ（登録商標）アプリを利用できる状態になり、ユーザーは、アプリケーション管理部１９１４が提供するＪａｖａ（登録商標）アプリケーション起動画面から起動するＪａｖａ（登録商標）アプリを選択することにより、仮想マシンを起動し、Ｊａｖａ（登録商標）アプリを実行することができる。

【０１９１】

次のようにしてＪａｖａ（登録商標）アプリを実行することも可能である。次に説明する方法は、図２９に示した構成において、Ｊａｖａ（登録商標）アプリをＷｅｂサーバからロードして実行する例である。以下の処理は、アプリケーション管理部１９１４に含まれるローダー（プログラム）により実行される。なお、ローダー自身はＪａｖａ（登録商標）アプリとして実装してもよいし、例えばＣ言語で実装してもよい。

【０１９２】

図３８に、図２９に示した構成におけるＪａｖａ（登録商標）アプリのアップロードからダウンロードまでの概念図を示す。このように、Ｊａｖａ（登録商標）開発環境において、開発、テストされたＪａｖａ（登録商標）アプリをＷｅｂサーバにアップロードする（ステップ１）。複合機のローダーは、Ｗｅｂサーバにアクセスし（ステップ２）、ユーザ所望のＪａｖａ（登録商標）アプリをダウンロードし（ステップ３）、実行する。

【０１９３】

次に、図３９のフローチャートを参照してローダーの処理手順について説明する。

【０１９４】

ユーザが複合機におけるアプリ切り替えのための所定のボタンを押すことにより、ローダーはＪａｖａ（登録商標）アプリが既にロードされているか否かをチェックし（ステップ１１）、ロードされていればＪａｖａ（登録商標）アプリを実行する（ステップ１２）。そして、Ｊａｖａ（登録商標）アプリユーザーインタフェースが表示される。ロードされていなければローダーの画面が表示される

（ステップ１３）。なお、Ｊａｖａ（登録商標）アプリの実行が終了した場合にはローダーの画面が表示される。また、Ｊａｖａ（登録商標）アプリのユーザーインタフェースが表示されているときに、所定のコードを入力して所定のボタンを押すことにより、ローダーの画面を表示することができる。

【０１９５】

図４０に、複合機に表示されるローダーの画面を示す。ローダーには、現在ロードされているＪａｖａ（登録商標）アプリがあればそのＪａｖａ（登録商標）アプリと、ローダーがＪａｖａ（登録商標）アプリの更新をＷｅｂサーバに対してチェックする頻度が表示される。

【０１９６】

ローダーの画面において、アプリのロードのボタンに触れると、図４１に示すアプリロードウィンドウが表示される（ステップ１４）。ここでＪａｖａ（登録商標）アプリのＵＲＬを入力する。ＵＲＬを入力するには、ＵＲＬボタンに触れ、ソフトキーボードを表示させ、ソフトキーボードからタイプすることによりＵＲＬを入力する。また、ＵＲＬボタンに触れることにより、図４２に示すようなＪａｖａ（登録商標）アプリのＵＲＬのリストを表示し、そのリストの中から所望のＪａｖａ（登録商標）アプリを選択するようにしてもよい。また、必要に応じて、図４１に示すクラス名の欄に、選択したＪａｖａ（登録商標）アプリ（Ｊａｒファイル）の中のｍａｉｎルーチンを持つクラス名を入力する。

【０１９７】

その後、図４１に示す画面のＯＫボタンに触れることにより、Ｊａｖａ（登録商標）アプリがＷｅｂサーバからダウンロードされ（ステップ１５）、複合機で実行される。

【０１９８】

ローダーは自動的に、指定された間隔でＷｅｂサーバをチェックし、新たなバージョンのＪａｖａ（登録商標）アプリがＷｅｂサーバに格納されているか否かを確認する。これは、現在動作しているＪａｖａ（登録商標）アプリの日付とタイムスタンプを、Ｗｅｂサーバ上のものと比較することにより行う。新たなバージョンのＪａｖａ（登録商標）アプリが発見された場合には、現在作動している

Ｊａｖａ（登録商標）アプリは削除され、新しいバージョンのＪａｖａ（登録商標）アプリがダウンロードされる。

【０１９９】

上記の更新間隔を変更するには、図４０に示す画面上の更新間隔ボタンに触れることにより、図４３に示す画面が表示される（ステップ１６）。この画面の中で矢印キーを用いて時間か分かを選択し、オペレーションパネルの数字キーを用いて新たな時間間隔を入力する。

【０２００】

複合機からＪａｖａ（登録商標）アプリを削除する場合には図４０に示す画面のアンロードのボタンに触れることにより、図４４に示す画面を表示する（ステップ１７）。ここで“ＯＫ”ボタンに触れることによりＪａｖａ（登録商標）アプリが削除される。

【０２０１】

図４０の画面におけるＥｘｉｔボタンに触れることによりローダーが終了する。

【０２０２】

さて、本実施の形態では、Ｊａｖａ（登録商標）アプリを連結することにより、カスタマイズプログラムを作成することが可能である。この処理の流れについて図４５を参照して説明する。なお、カスタマイズプログラムを作成するプログラム自体もＪａｖａ（登録商標）アプリであり、以下これをカスタマイズアプリと呼ぶ。なお、カスタマイズアプリは予め画像形成装置内に備えられていてもよい。

【０２０３】

まず、操作表示部の画面を図４０のローダ画面に切り替え、カスタマイズアプリをロードし、実行する。

【０２０４】

カスタマイズアプリは図４５の画面１を表示する。画面１には、紙（paper）のオブジェクトだけが表示される。なお、紙のオブジェクト以外のオブジェクトを入れることもできる。オブジェクトは“入れる”ボタン、“取り出す”ボタン

により出し入れをすることができる。また、画面１において、紙のオブジェクトには読み書き許可属性などの属性を“読書許可”ボタンにより設定することができる。

【０２０５】

画面１においてpaperに触れると、画面２に示すように、paperオブジェクトに接続されるオブジェクトのリストが表示される。ここでscanを選択すると、画面３に遷移し、さらにscanに触れることにより、scanオブジェクトに接続されるオブジェクトのリストが画面４のように表示される。画面４においてmailを選択すると画面５に遷移する。

【０２０６】

その後、“設定”ボタンに触れることにより、スキャンして、その画像ファイルをメールにて送信するという処理を行うカスタマイズプログラムが生成される。また、画面６に示すようにキーがアサインされる。複合機でこのキーを操作することにより、上記のカスタマイズプログラムが実行される。このカスタマイズプログラムは、連結されたオブジェクトのクラスを順次実行するように記述されたプログラムである。

【０２０７】

なお、本実施の形態のようにＷｅｂサーバにアプリを格納しておき、Ｗｅｂサーバから複合機に所望のアプリをダウンロードすることや、アプリを連結してカスタマイズすることは、他の実施の形態にも適用可能である。例えば、ＶＢインタープリタを用いた前述した構成において、本実施の形態と同様の機能のローダーを備えることにより、本実施の形態と同様にサーバからアプリをダウンロードできる。

【０２０８】

なお、これまでに説明した各サービスプログラム、アプリケーションプログラムは、ＩＣカード等の記録媒体に格納し、そこから画像形成装置に格納することができる。また、ネットワークを介して外部のサーバから画像形成装置に格納することもできる。

【０２０９】

上記のように、本実施の形態によれば、Ｊａｖａ（登録商標）環境を用いることにより効率良くアプリを開発することが可能となる。

本発明は、上記の実施の形態に限定されることなく、特許請求の範囲内で種々変更・応用が可能である。

【発明の効果】

上記のように本発明によれば、アプリケーションを容易に実行可能な画像形成装置およびアプリケーション実行方法を得ることができる。

【図面の簡単な説明】

【図１】

実施の形態１にかかる複合機の主要構成およびネットワーク構成を示すブロック図である。

【図２】

実施の形態１の複合機の機能的構成を示すブロック図である。

【図３】

実施の形態１にかかる複合機のハードウェア構成を示すブロック図である。

【図４】

リクエストメッセージを受信した複合機における処理の手順を示すフローチャートである。

【図５】

ｄｅｂｕｇ．ｃｇｉの処理の手順を示すフローチャートである。

【図６】

プログラミング画面において、保存ボタン、実行設定ボタン、テスト実行ボタンのいずれかのクリック操作が行われた場合のＷＥＢアプリの処理の手順を示すフローチャートである。

【図７】

プログラミング画面の内容の一例を示す説明図である。

【図８】

プログラム入力フィールドに入力されるカスタマイズプログラムの一例を示す説明図である。

【図９】

カスタマイズディレクトリに格納されたカスタマイズプログラムの名称の一例を示す説明図である。

【図１０】

キー対応付けテーブルの内容の一例を示す説明図である。

【図１１】

複合機の電源投入が行われてから、カスタマイズプログラムの実行が可能となるまでの一連の処理の手順を示すフローチャートである。

【図１２】

起動設定ファイルの一例を示す説明図である。

【図１３】

実施の形態２にかかる複合機における主要構成を示すブロック図である。

【図１４】

プログラミング画面の一例を示す説明図である。

【図１５】

実施の形態３にかかる複合機の機能的構成を示すブロック図である。

【図１６】

複合機の電源投入が行われてから、カスタマイズプログラムとボタン（キー）との割り付けが完了するまでの一連の処理の手順を示すフローチャートである。

【図１７】

起動設定ファイルの内容の一例を示す説明図である。

【図１８】

キー割り付け設定画面の一例を示す説明図である。

【図１９】

実施の形態４にかかる複合機の主要構成およびネットワーク構成を示すブロック図である。

【図２０】

実施の形態４にかかる複合機の機能的構成を示すブロック図である。

【図２１】

起動設定ファイルの内容の一例を示す説明図である。

【図２２】

実施の形態５にかかる複合機の機能的構成を示すブロック図である。

【図２３】

ＰＣ２００上でＶｉｓｕａｌ　Ｂａｓｉｃ開発環境を用いてＶＢアプリを作成している場面を示す図である。

【図２４】

ＶＢインタプリタにより実行されるＶＢアプリの例を示す図である。

【図２５】

ＶＢインタプリタにより実行されるＶＢアプリの例を示す図である。

【図２６】

実施の形態６にかかる複合機の機能的構成を示すブロック図である。

【図２７】

Ｊａｖａ（登録商標）開発環境とＪａｖａ（登録商標）実行環境を示す図である。

【図２８】

Ｊａｖａ（登録商標）実行環境の構成の例を示す図である。

【図２９】

ＷｅｂサーバからＪａｖａ（登録商標）アプリをダウンロードする構成を示す図である。

【図３０】

Ｊａｖａ（登録商標）アプリ作成の手順を示す図である。

【図３１】

エミュレータの他の実装例を示す図である。

【図３２】

エミュレータによりＰＣに表示されるオペレーションパネルの画面を示す図である。

【図３３】

エミュレータがローダーとして動作する場合に表示される画面を示す図である

。

【図３４】

操作パネルクラスの階層構成を示す図である。

【図３５】

ウィンドウの作成を説明するための図である。

【図３６】

サンプルプログラムを示す図である。

【図３７】

サンプルプログラムの実行結果を示す図である。

【図３８】

図２９に示した構成におけるＪａｖａ（登録商標）アプリのアップロードからダウンロードまでの概念図である。

【図３９】

ローダーの処理手順を示すフローチャートである。

【図４０】

複合機に表示されるローダーの画面を示す図である。

【図４１】

アプリをロードするための画面である。

【図４２】

Ｊａｖａ（登録商標）アプリのＵＲＬのリストである。

【図４３】

更新間隔を変更するための画面である。

【図４４】

アプリをアンロードするための画面である。

【図４５】

Ｊａｖａ（登録商標）アプリを連結することにより、カスタマイズプログラムを作成する手順を説明するための図である。

【符号の説明】

１００，１１００，１３００，１７００，１８００，１９００　複合機

１０１　白黒ラインプリンタ
１０２　カラーラインプリンタ
１０３　ハードディスク装置（ＨＤＤ）
１０４　ハードウェアリソース
１０５　共有メモリ
１０６　ｈｔｔｐｄ
１１０　ソフトウェア群
１１１　プリンタアプリ
１１２　コピーアプリ
１１３　ファックスアプリ
１１４　スキャナアプリ
１１５　ネットファイルアプリ
１１６　工程検査アプリ
１１７　外部アプリ
１２０　プラットホーム
１２１　汎用ＯＳ
１２２　ＳＣＳ
１２３　ＳＲＭ
１２４　ＥＣＳ
１２５　ＭＣＳ
１２６　ＯＣＳ
１２７　ＦＣＳ
１２８　ＮＣＳ
１２９　複合機初期化部
１３０　アプリケーション
１３１　プログラム起動部
１３２，１１３２　プログラミングサービス
１３３　オペパネ登録部
１３４　インタプリタ（シェル）

１３５ カーネル

２００ ＰＣ２０１，１１０１プログラミング画面

２１１ 起動設定ファイル

２１２ カスタマイズディレクトリ

２２０ インターネット

１７３４，　１８０１ ＶＢインタプリタ

１９０１　Ｊａｖａ（登録商標）実行環境

１９１１、１９２１　クラスライブラリ

１９１２、１９２２　仮想マシン

１９１４、１９２４　アプリケーション管理部

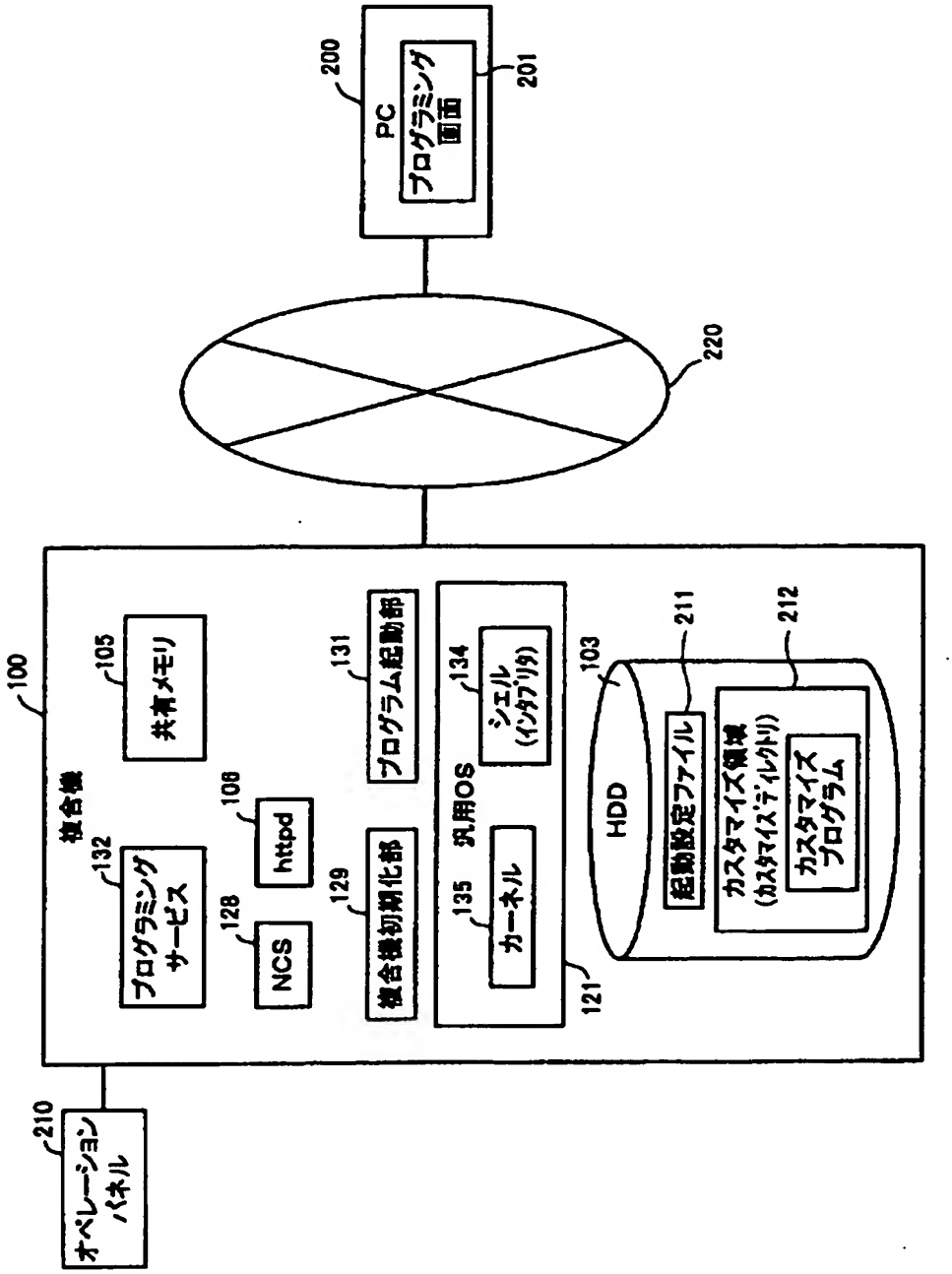
１９２３　エミュレータ

１９２５　Ｊａｖａ（登録商標）コンパイラ

【書類名】　図面

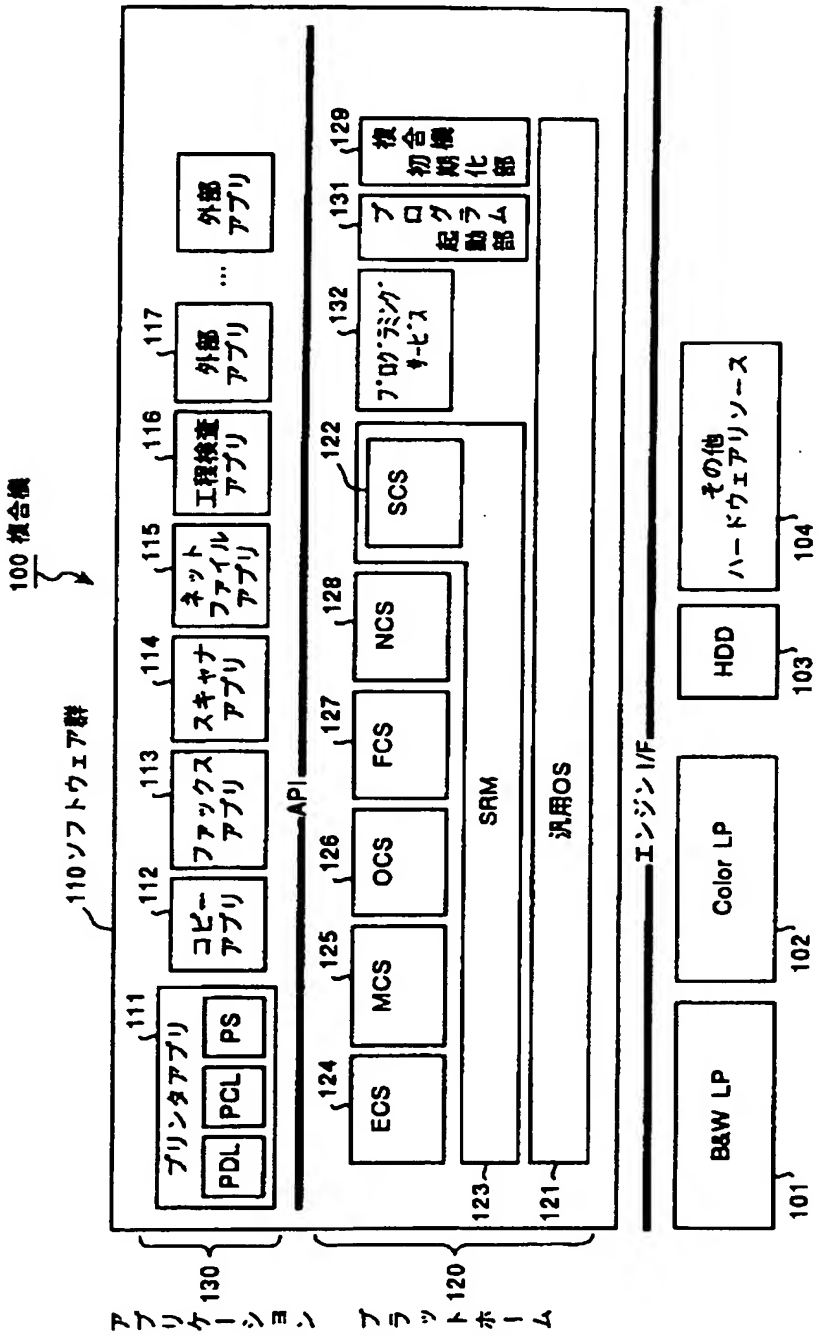
【図１】

実施の形態1にかかる複合機の主要構成および
ネットワーク構成を示すブロック図



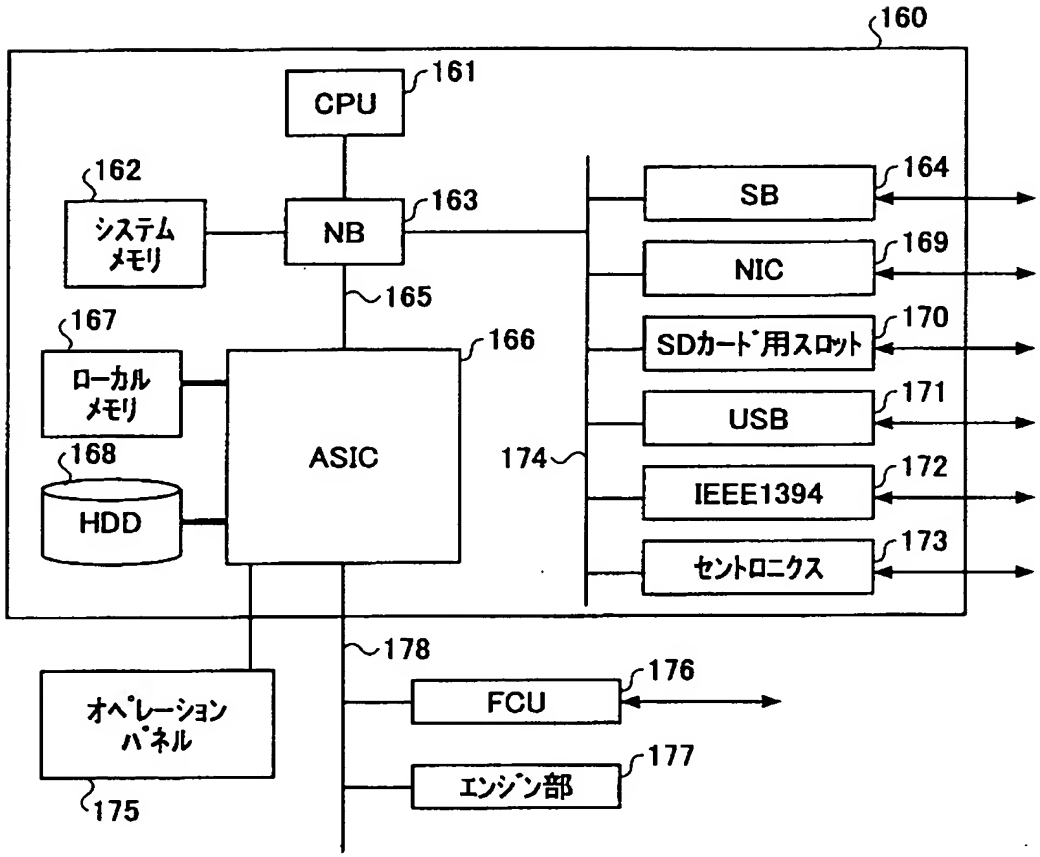
【図２】

実施の形態1の複合機の機能的構成を示すブロック図



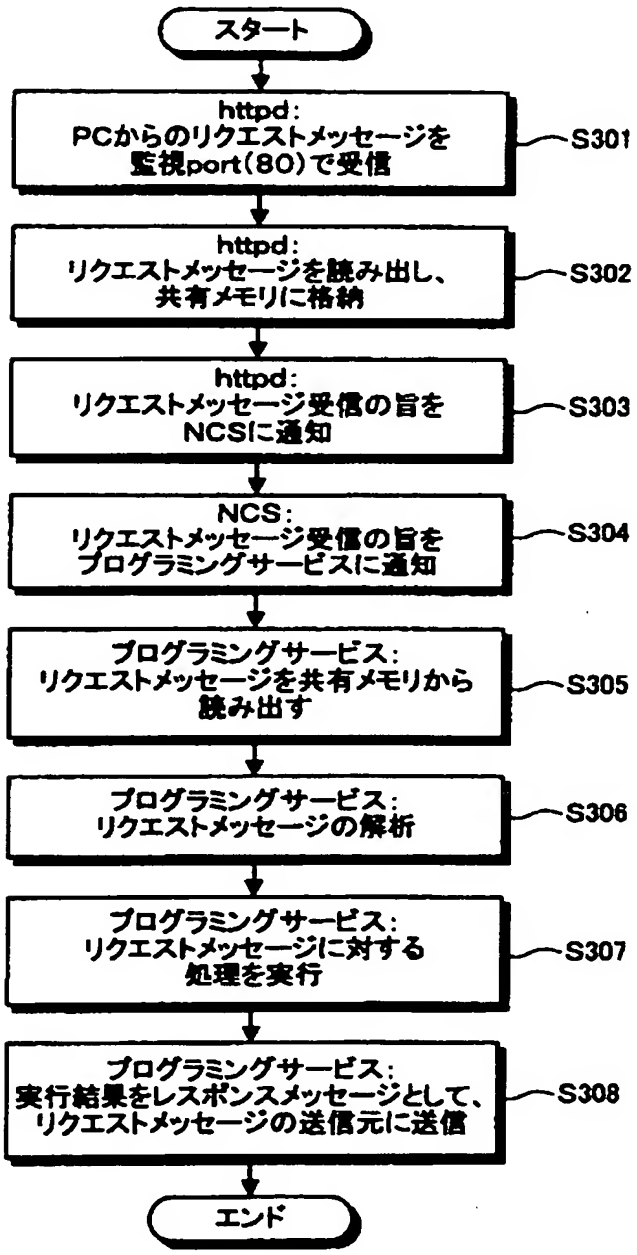
【図３】

実施の形態1にかかる複合機の
ハードウェア構成を示すブロック図



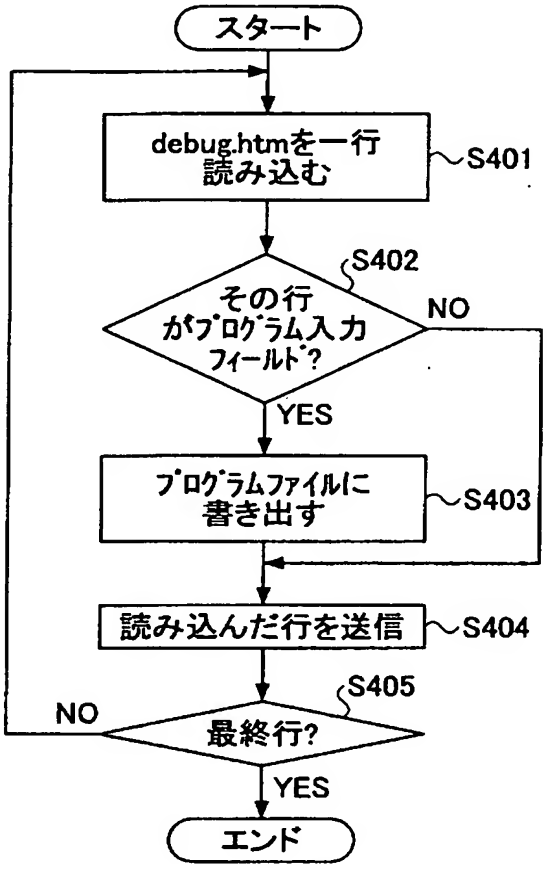
【図４】

リクエストメッセージを受信した複合機における
処理の手順を示すフローチャート



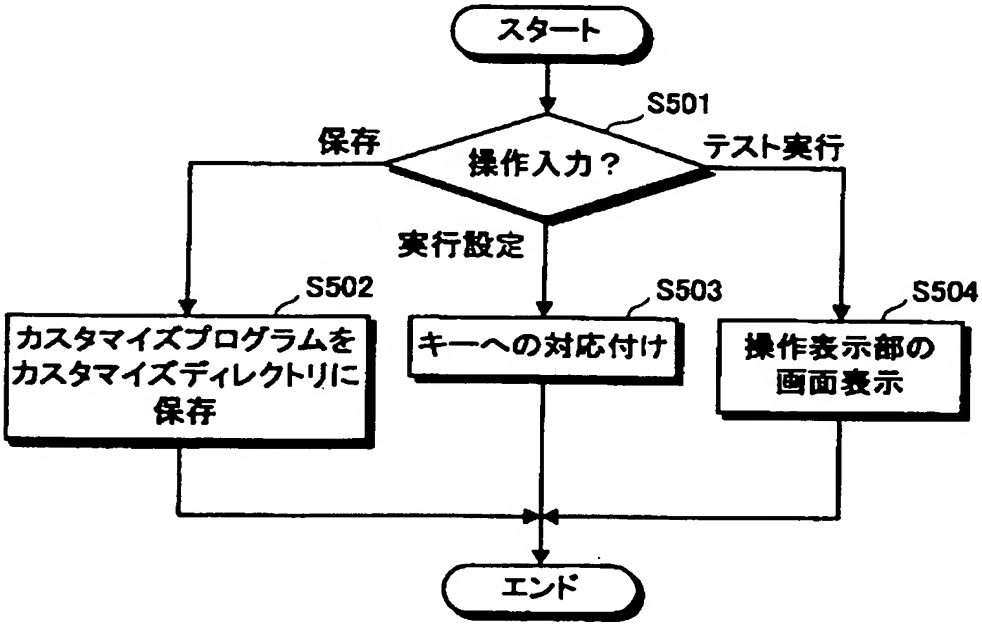
【図５】

debug. cgiの処理の手順を示すフローチャート



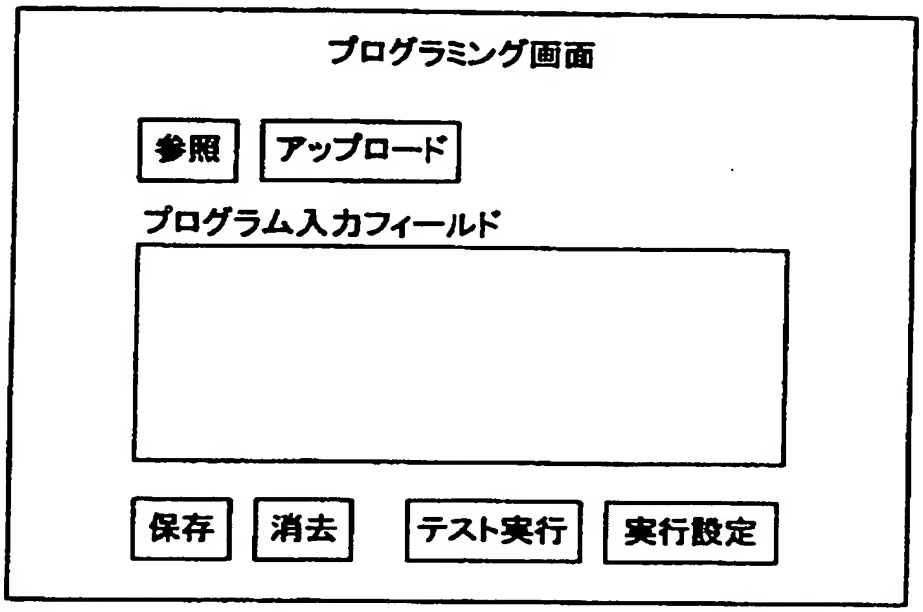
【図６】

プログラミング画面において、保存ボタン、実行設定ボタン、
テスト実行ボタンのいずれかのクリック操作が行われた場合の
WEBアプリの処理の手順を示すフローチャート



【図７】

プロムラミング画面の内容の一例を示す説明図



【図８】

プロムラミング入力フィールドに入力される
カスタマイズプログラムの一例を示す説明図

```
# ocr shell script
#
scanimage -out /hdd/ts/tmp/tmpimg.tif
decomp -in /hdd/ts/tmp/tmpimg.tif -out /hdd/ts/tmp/tmpimg.bmp
ocr -in /hdd/ts/tmp/tmpimg.bmp -out /hdd/ts/tmp/ocrresult.txt
mail xxxxx@yy.zz.co.jp
```

【図９】

カスタマイズディレクトリに格納された
カスタマイズプログラムの名称の一例を示す説明図

カスタマイズディレクトリ／ファイル名

hdd／xxx／opepane／shell1
shell2
shell3

【図１０】

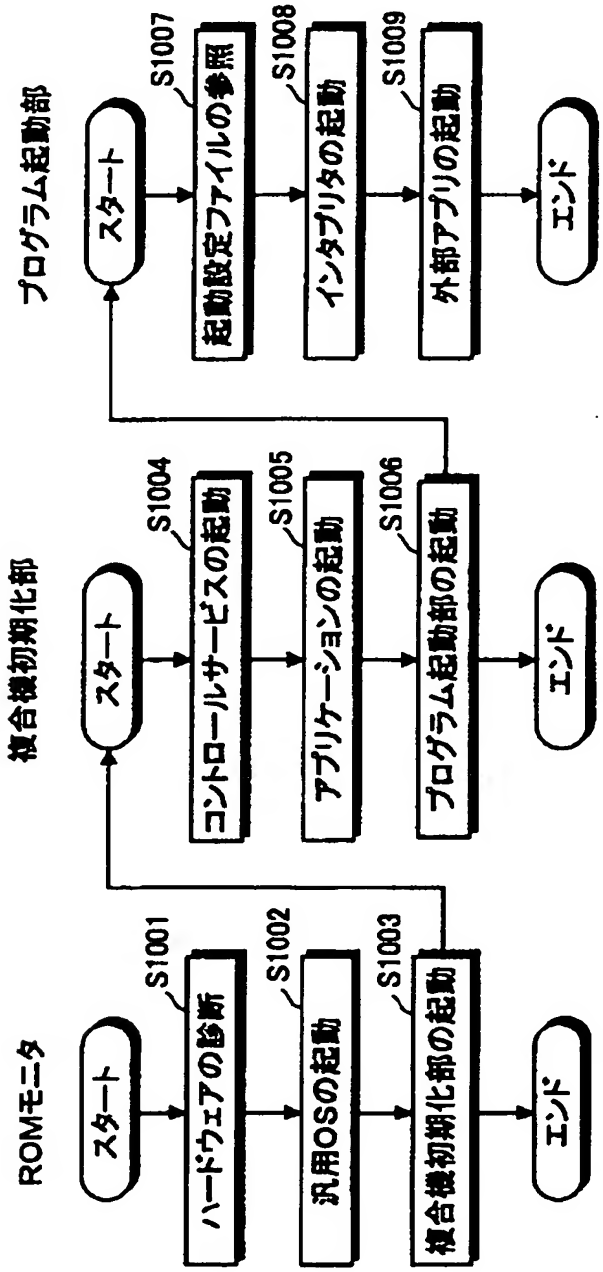
キー対応付けテーブルの内容の一例を示す説明図

キー対応付けテーブル

キーコード：	カスタマイズプログラム
501 ：	shell1
502 ：	shell2
503 ：	shell3

【図１１】

複合機の電源投入が行われてから、カスタマイズプログラムの実行が可能となるまでの一連の処理の手順を示すフローチャート



【図１２】

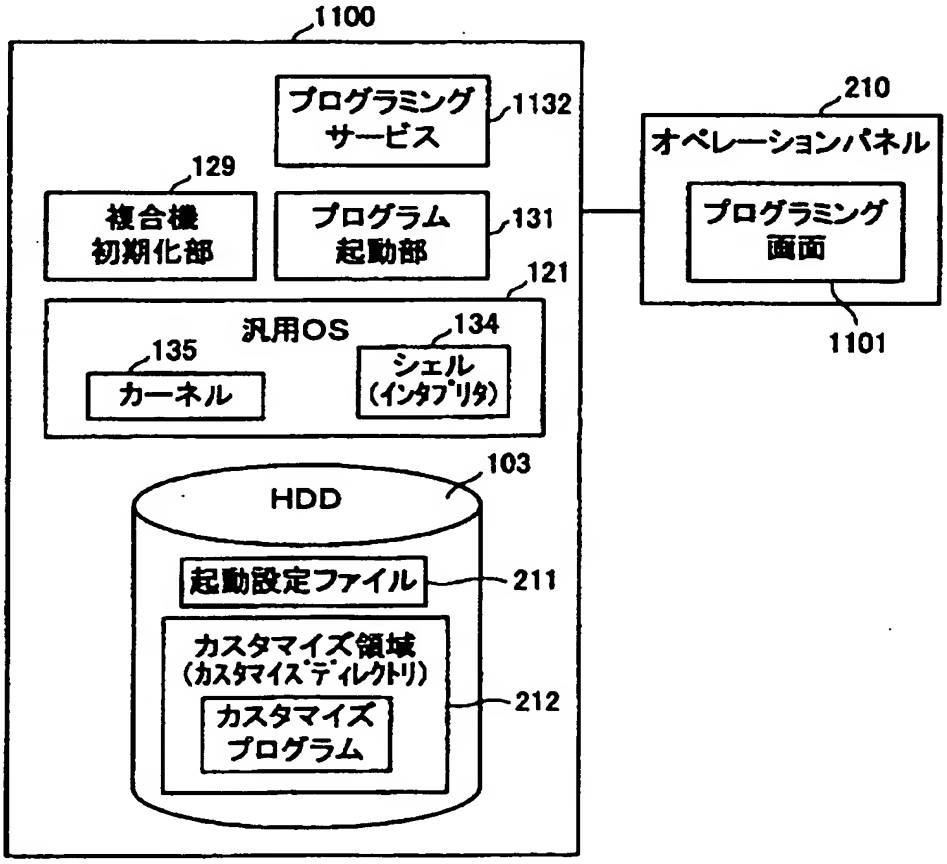
起動設定ファイルの一例を示す説明図

起動設定ファイル

プログラム名称	インタプリタ(シェル)
プログラム名称	XXXアプリ
：	

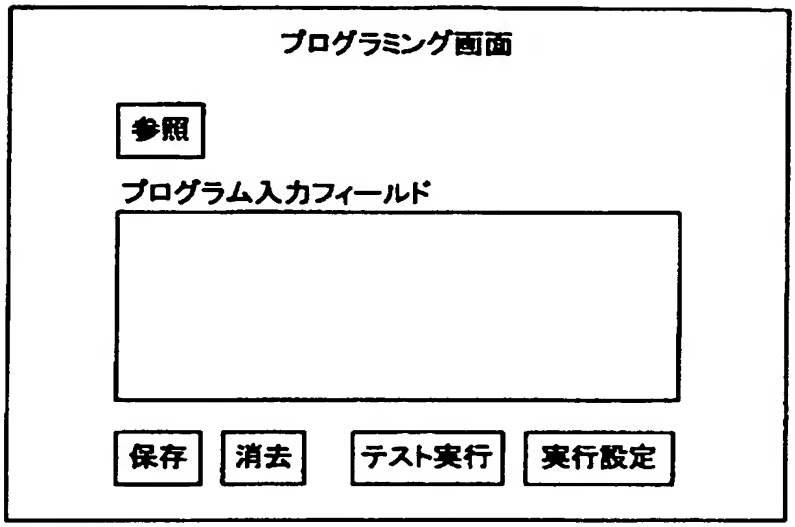
【図１３】

実施の形態2にかかる複合機における主要構成を示すブロック図



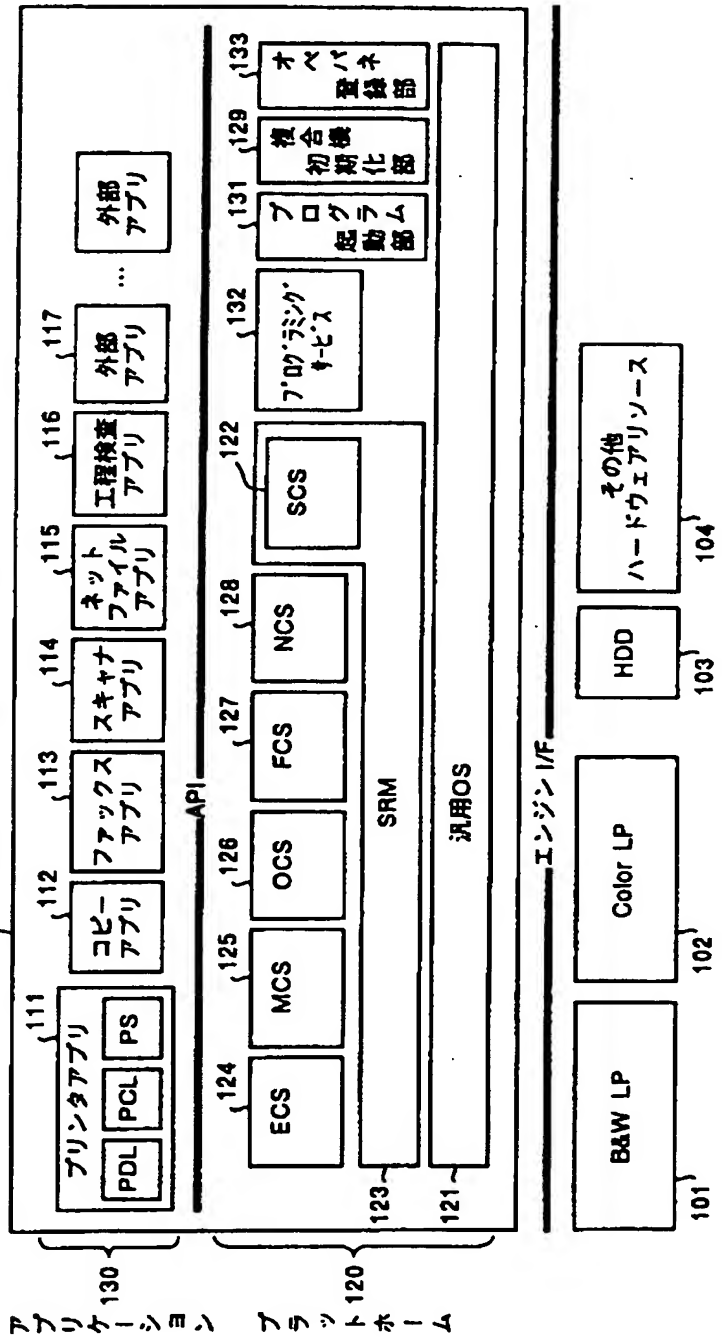
【図１４】

プロムラミング画面の一例を示す説明図



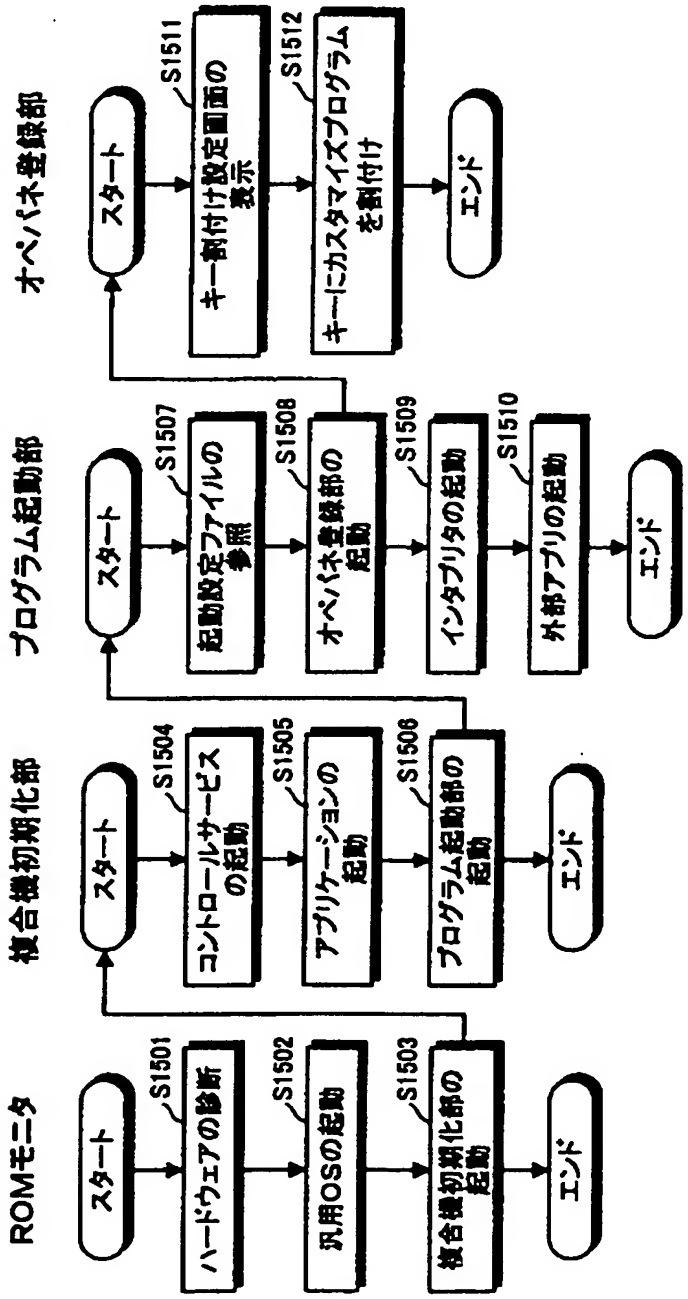
【図１５】

実施の形態3にかかる複合機の機能的構成を示すブロック図



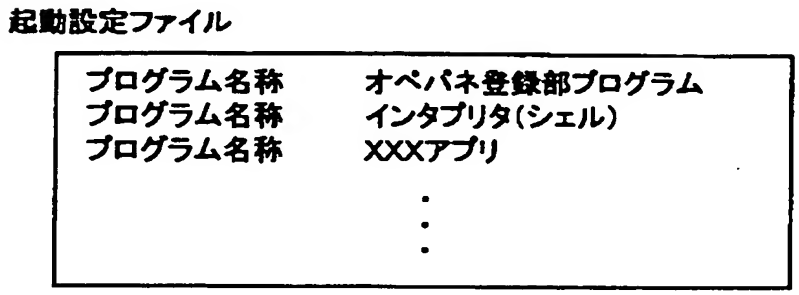
【図１６】

複合機の電源投入が行われてから、カスタマイズプログラムとボタン(キー)との割り付けが完了するまでの一連の処理の手順を示すフローチャート



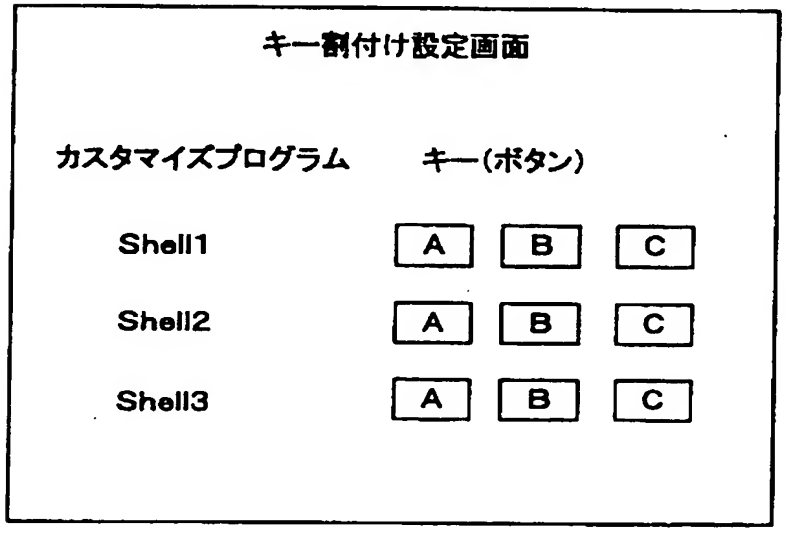
【図１７】

起動設定ファイルの内容の一例を示す説明図



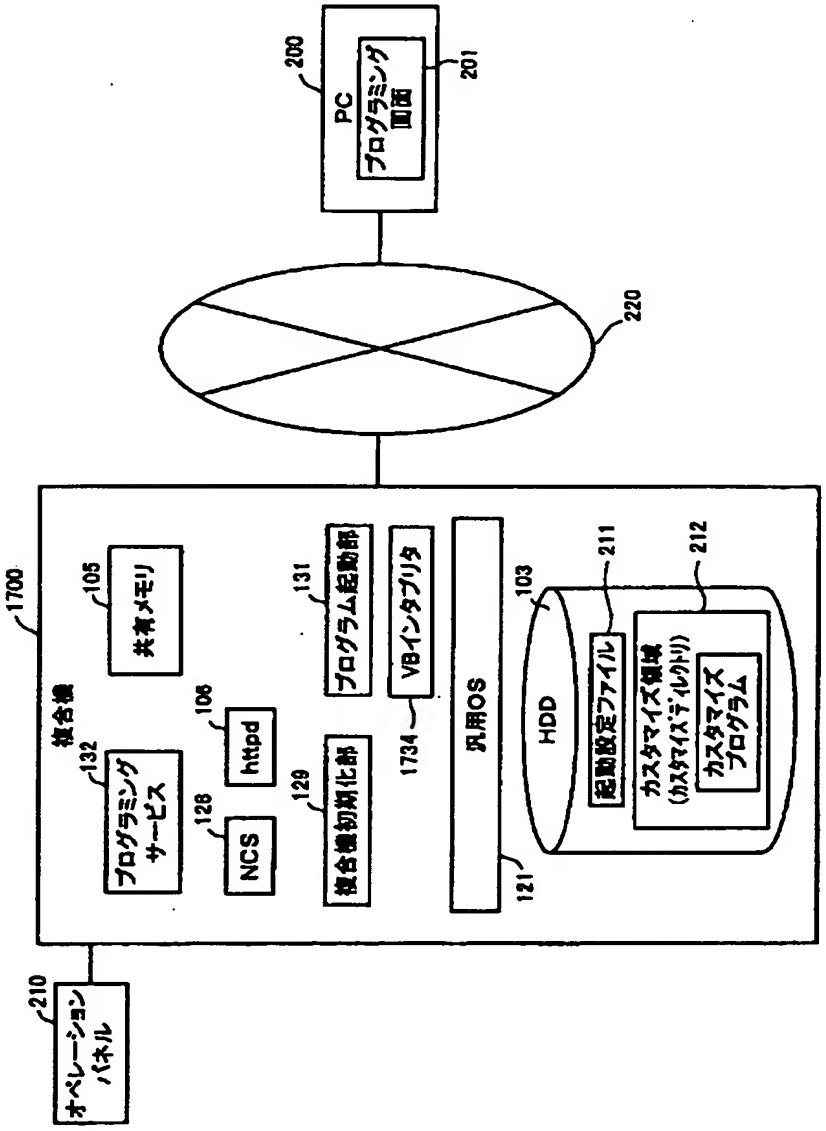
【図１８】

キー割り付け設定画面の一例を示す説明図



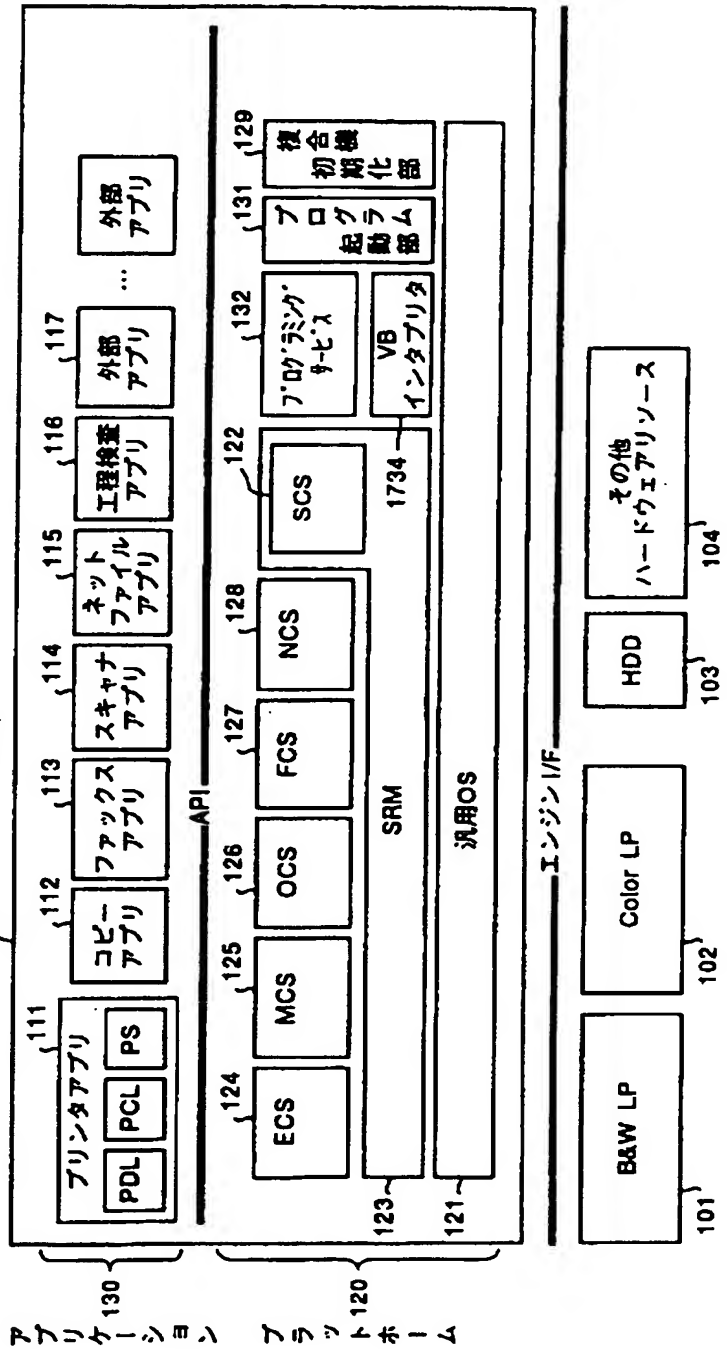
【図１９】

実施の形態4にかかる複合機の主要構成および
ネットワーク構成を示すブロック図



【図２０】

実施の形態4にかかる複合機の機能的構成を示すブロック図



【図２１】

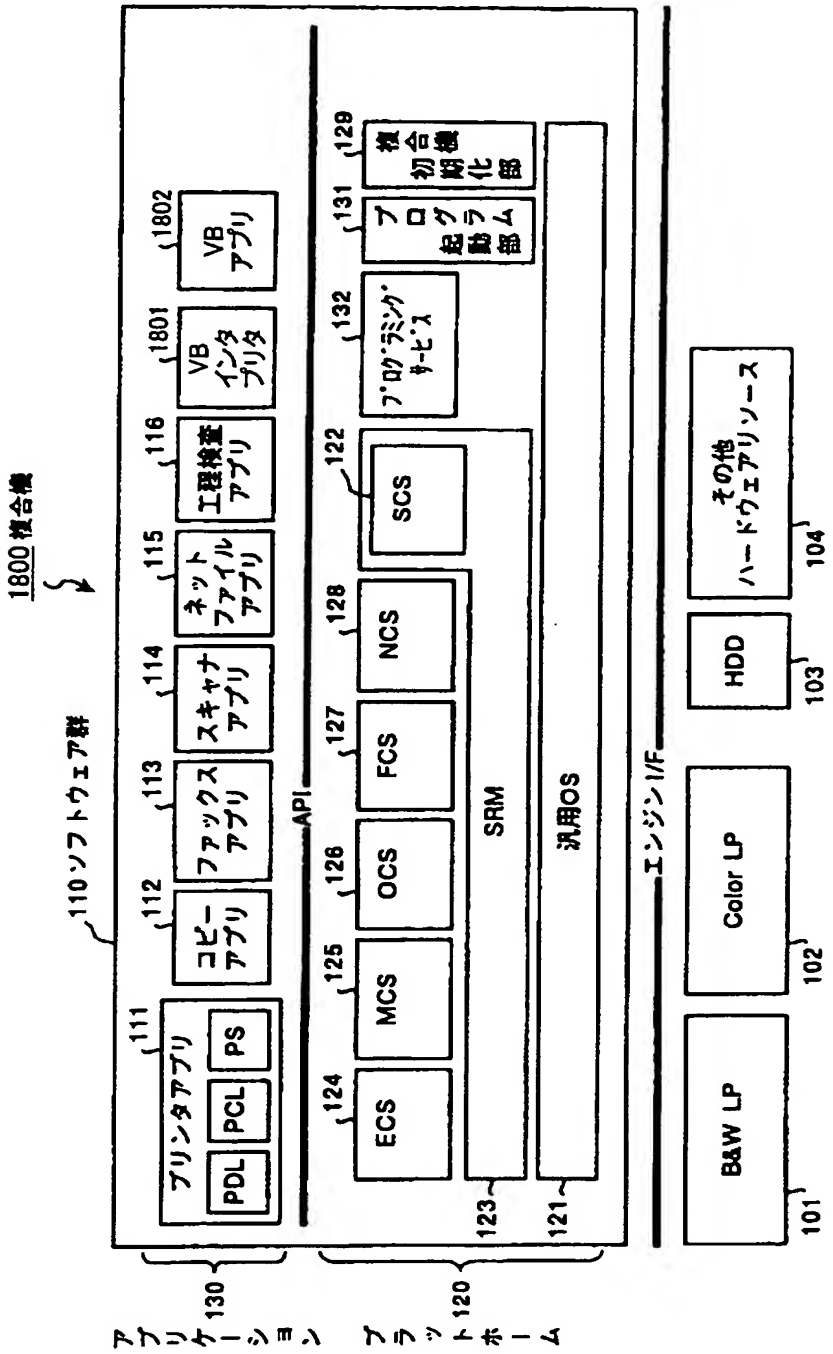
起動設定ファイルの内容の一例を示す説明図

起動設定ファイル

プログラム名称	VBインタプリタ
プログラム名称	XXXアプリ
：	

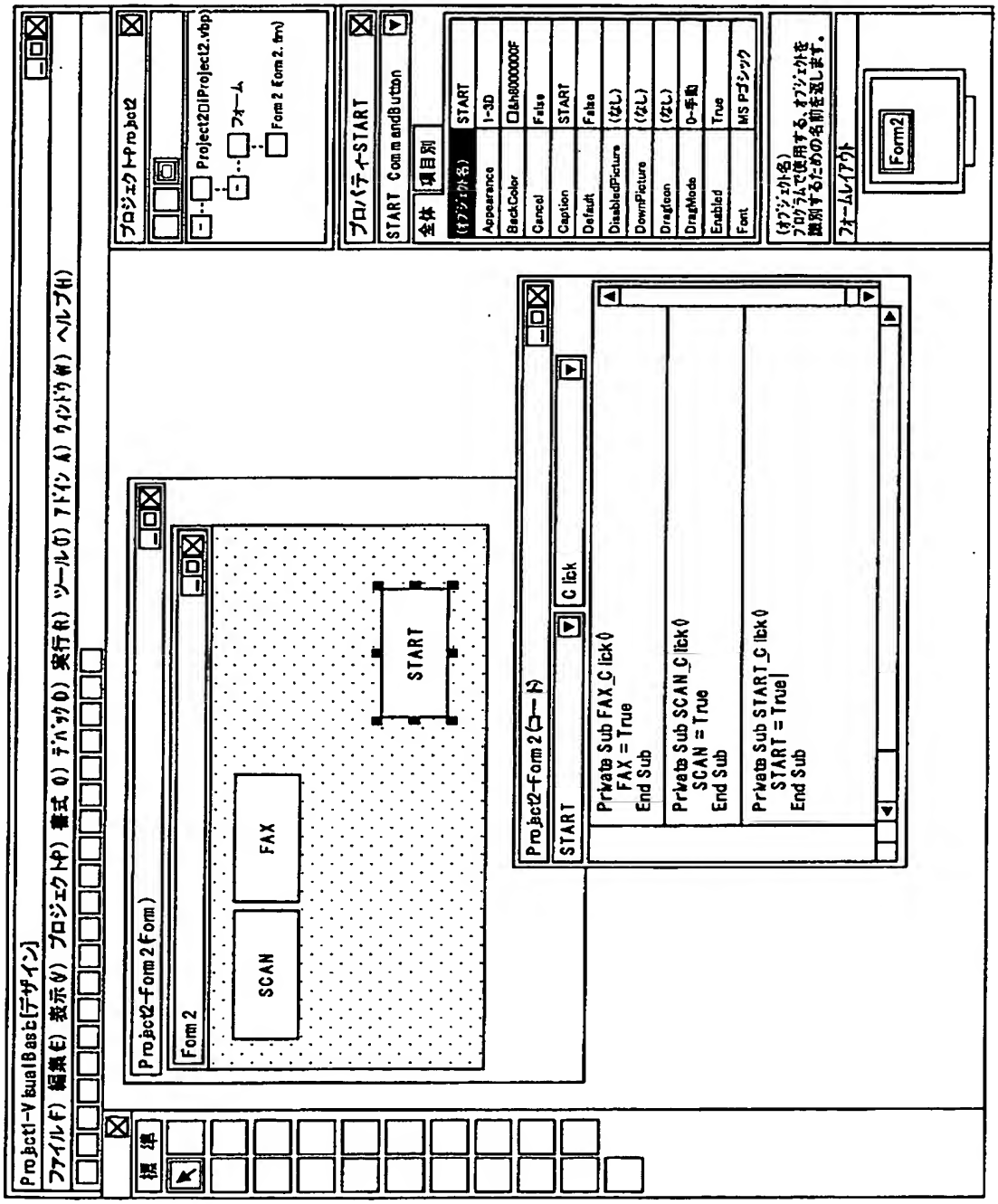
【図２２】

実施の形態5にかかる複合機の機能的構成を示すブロック図



【図２３】

PC200上でVisual Basic開発環境を用いて
VBアプリを作成している場面を示す図



【図２４】

VBインタプリタにより実行されるVBアプリの例を示す図

```
Begin VB. Form Form1
    Caption          =  "Form1"
    ClientHeight     =  3390
    ClientLeft       =  60
    ClientTop        =  345
    ClientWidth      =  6990
    LinkTopic        =  "Form1"
    ScaleHeight      =  3390
    ScaleWight       =  6990
    StartUpPosition  =  3 'Windows　の既定値
    Begin VB. CommandButton NEXTWIN
        Caption          =  "NEXTWIN"
        Height           =  735
        Left             =  480
        TabIndex         =  0
        Top              =  480
        Width            =  1575
    End
End
Attribute VB_Name = "Form1"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False

Private Sub NEXTWIN_Click()
   /*Form2 ウインドウへの切り替え  */
   call("change_diplay", "From2")
End Sub
```

【図２５】

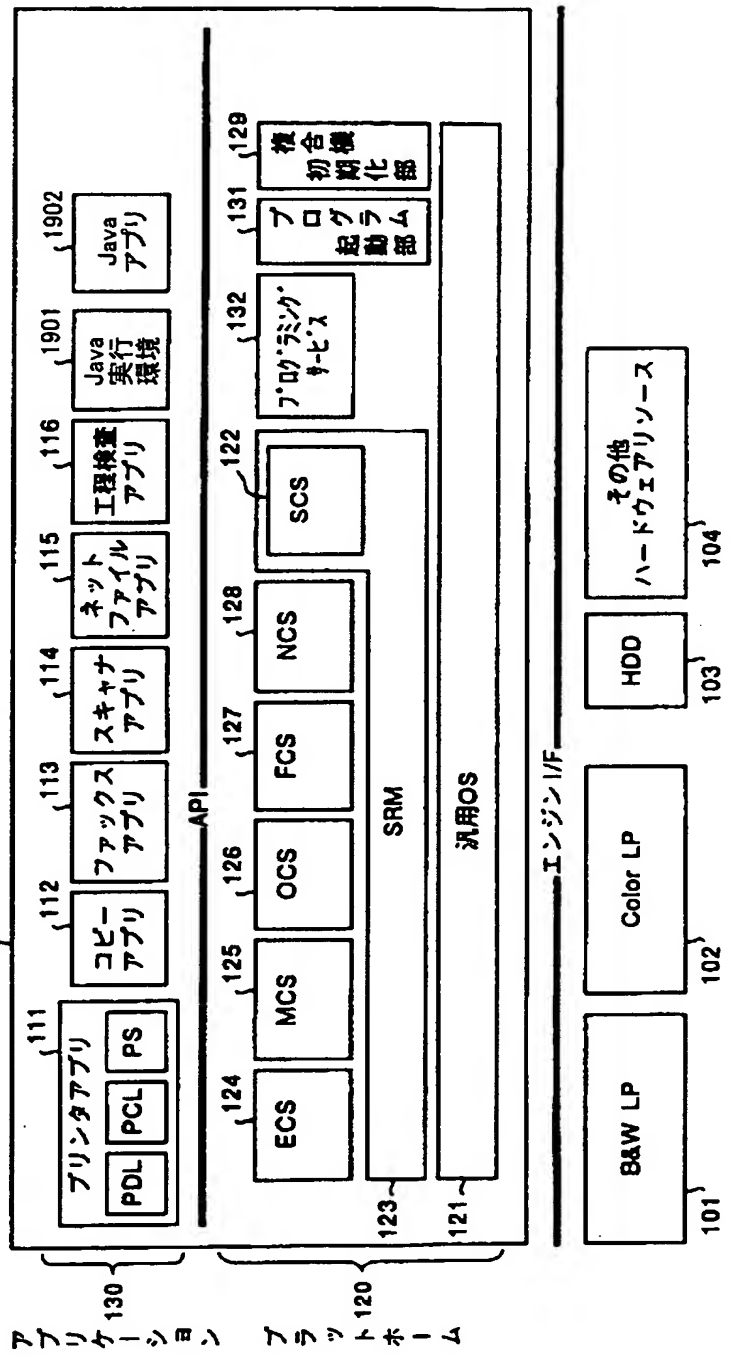
VBインタプリタにより実行されるVBアプリの例を示す図

```
Begin VB. Form Form2
      Caption              =   "Form2"
      ClientHeight         =   3390
      ClientLeft           =   60
      ClientTop            =   345
      ClientWidth          =   6990
      LinkTopic            =   "Form2"
      ScaleHeight          =   3390
      ScaleWight           =   6990
      StartUpPosition      =   3 'Windows  の既定値
      Begin VB. CommandButton START
            Caption              =   "START"
            Height               =   735
            Left                 =   4560
            TabIndex             =   2
            Top                  =   2040
            Width                =   1815
      End
      Begin VB. CommandButton FAX
            Caption              =   "FAX"
            Height               =   735
            Left                 =   2520
            TabIndex             =   1
            Top                  =   480
            Width                =   1455
      End
      Begin VB. CommandButton SCAN
            Caption              =   "SCAN"
            Height               =   735
            Left                 =   480
            TabIndex             =   0
            Top                  =   480
            Width                =   1575
      End
End
Attribute VB_Name = "Form1"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Private Sub FAX_Click()
    FAX = True
End Sub
Private Sub SCAN_Click()
    call("ScanStart", A4, ADF, TIFF, BINARY, "/work/tmpfile.tif")
    gwOpItemCreate(win, MESSAGE_ITEM,
                   ITEM_MESSAGE_X,          0,
                   ITEM_MESSAGE_Y,          12,
                   ITEM_MESSAGE_WIDTH,      100,
                   ITEM_MESSAGE_HEIGHT,     12,
                   ITEM_MESSAGE_CENTER_X,   0,
                   ITEM_MESSAGE_BLINK,      0,
                   ITEM_MESSAGE,            "文字列", 0,
                   ITEM_MESSAGE_FRONT,      FONT_12, 0,
                   0);
End Sub

Private Sub START_Click()
    START = True
End Sub
```

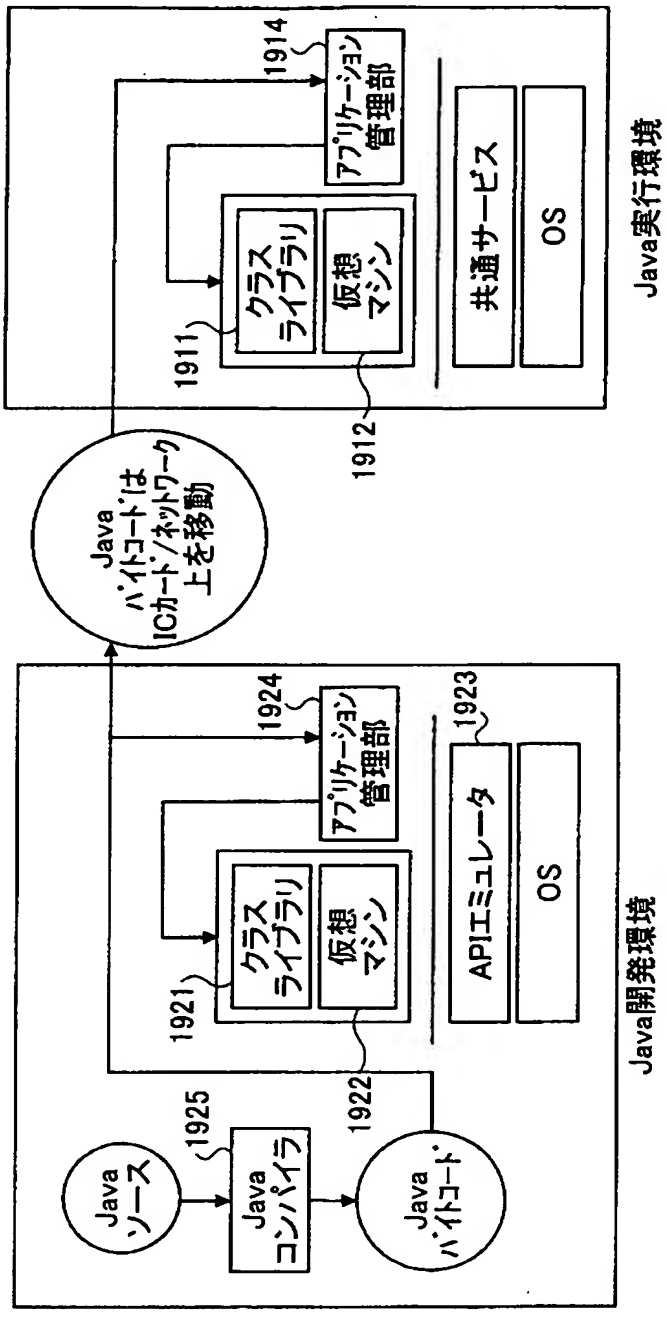
【図２６】

実施の形態6にかかる複合機の機能的構成を示すブロック図



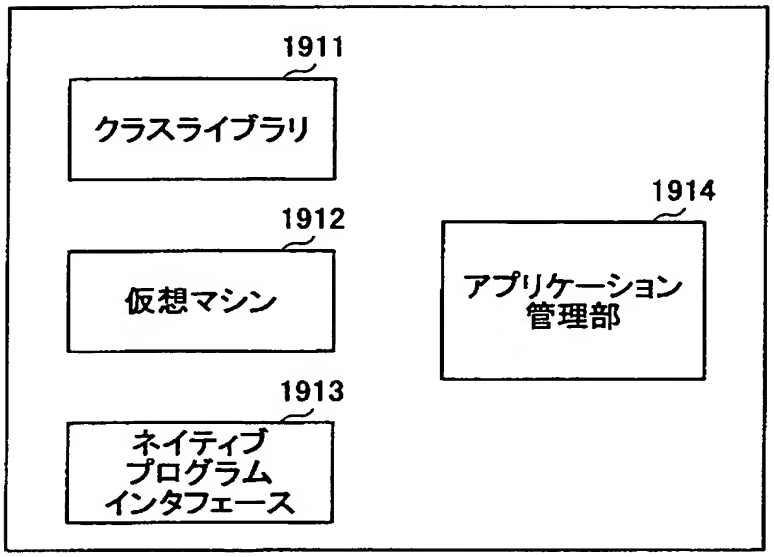
【図２７】

Java（登録商標）開発環境とJava実行環境を示す図



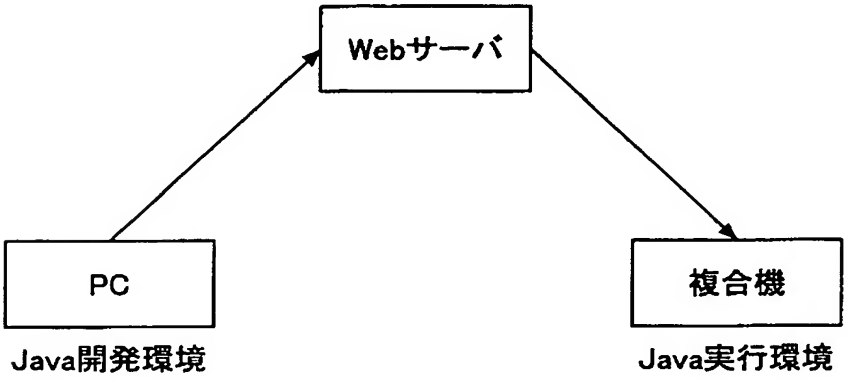
【図２８】

Java(登録商標)実行環境の構成の例を示す図



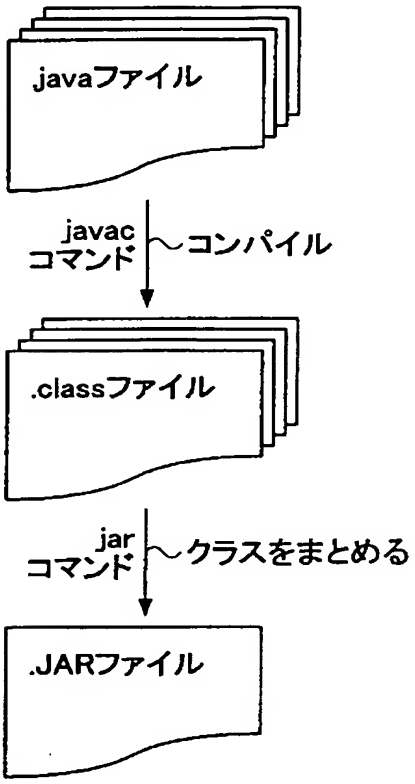
【図２９】

Ｗｅｂサーバから Ｊａｖａアプリをダウンロードする
構成を示す図



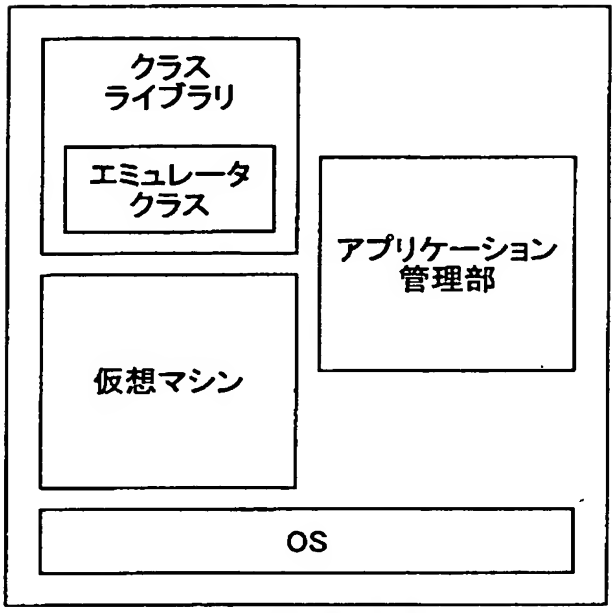
【図３０】

Javaアプリ作成の手順を示す図



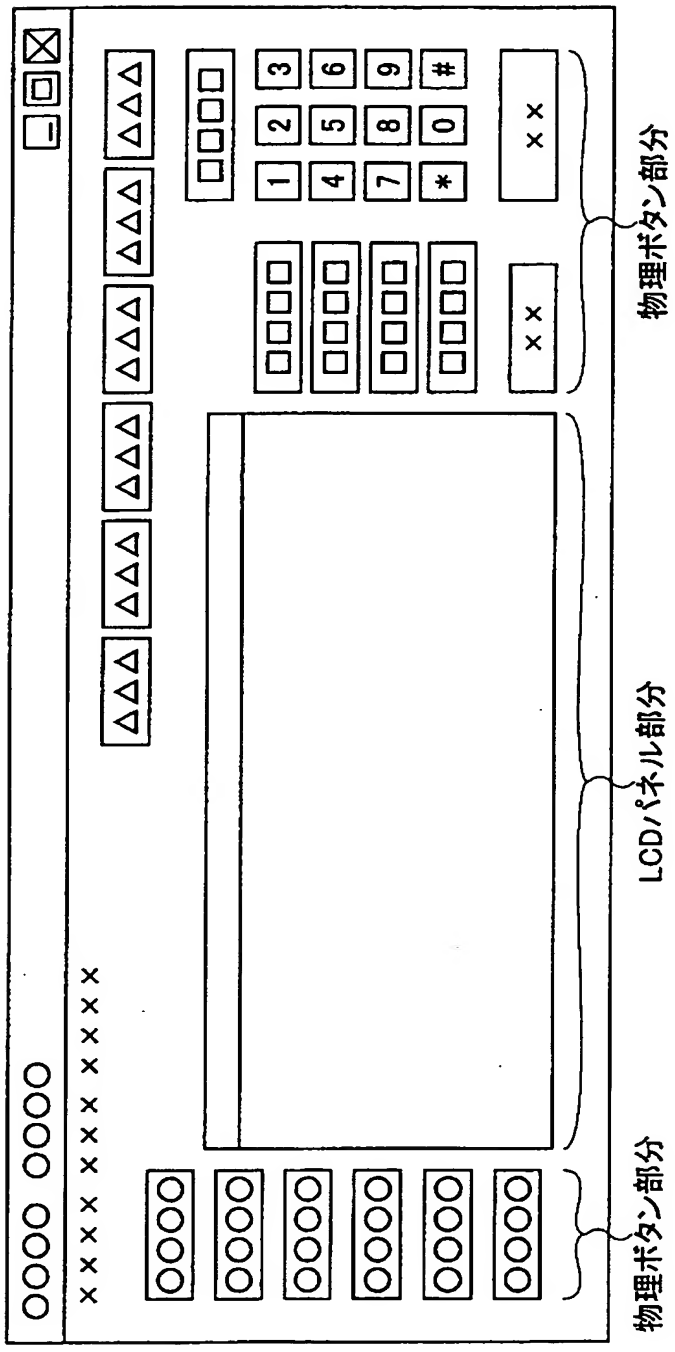
【図３１】

エミュレータの他の実装例を示す図



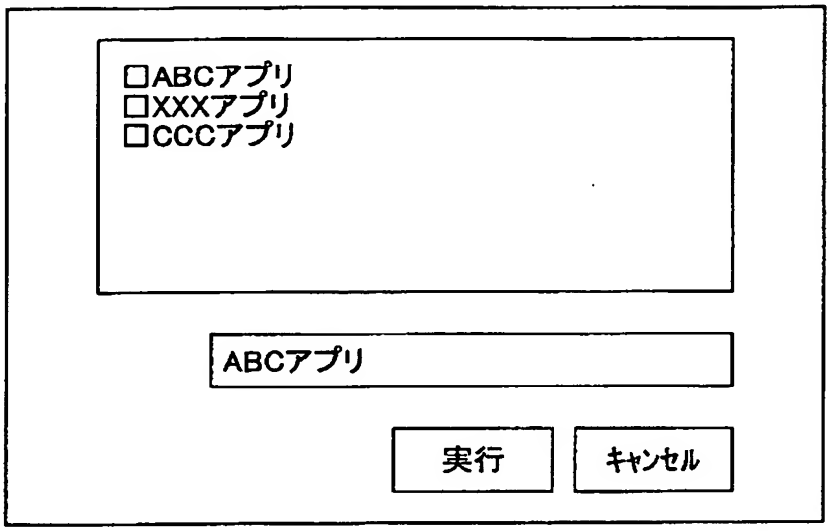
【図３２】

エミュレータによりPCに表示される
オペレーションパネルの画面を示す図



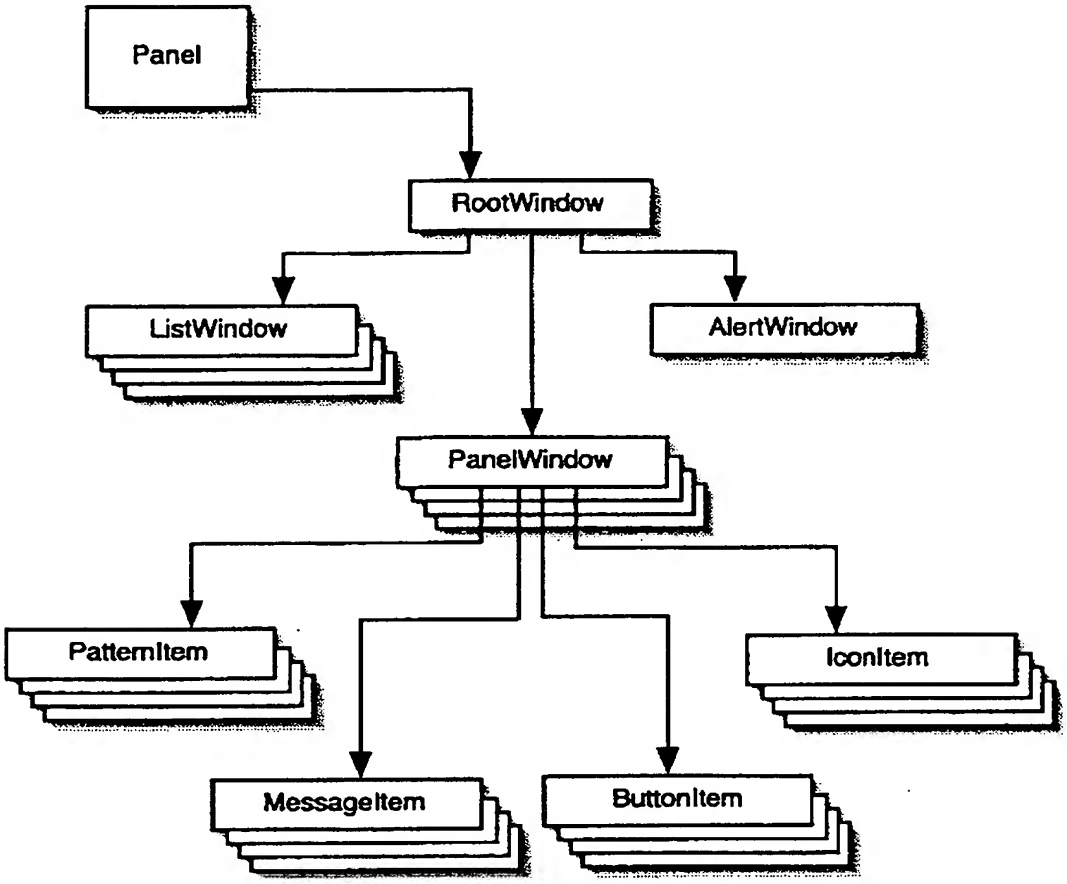
【図３３】

エミュレータがローダーとして動作する場合に表示される画面を示す図



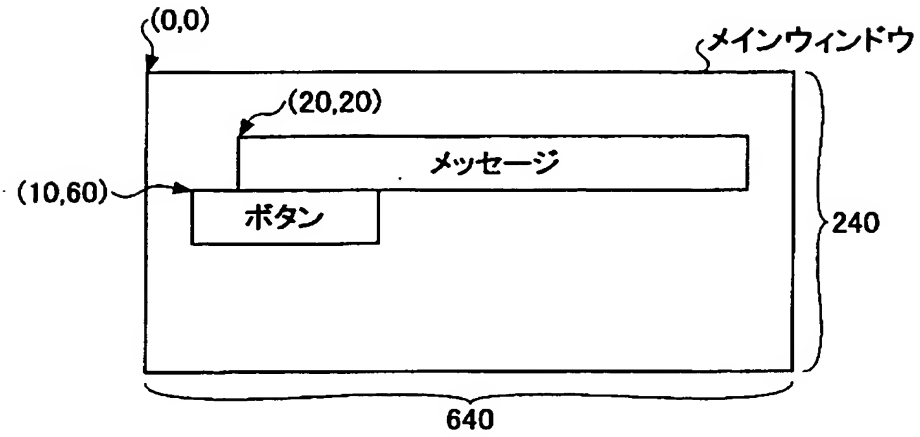
【図３４】

操作パネルクラスの階層構成を示す図



【図３５】

ウィンドウの作成を説明するための図



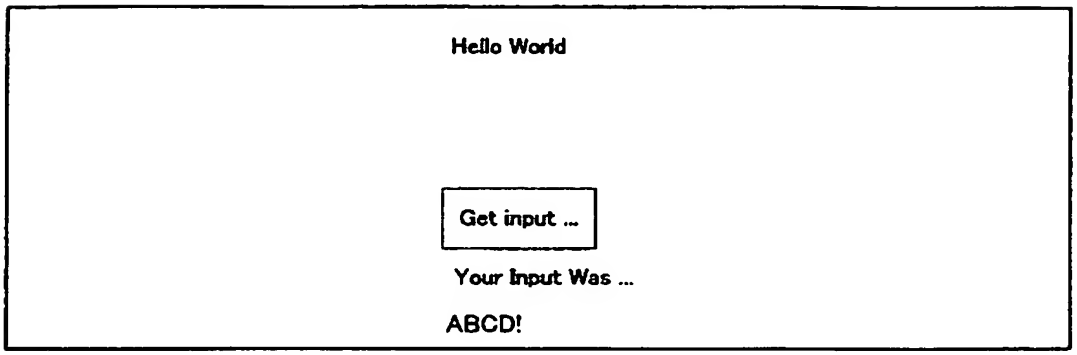
【図３６】

サンプルプログラムを示す図

```
import java.lang.*;
import com.hijke.gw.panel.*;                                    //←①
public class UserApp extends GWApp                              //←②
{
    public void main()
    {
        Panel panel = new Panel();
        PanelWindow mainwindow =
        new PanelWindow(panel.root());
        ItemString hello = new ItemString("Hello World");       //←③
        MessageItem salutation = new MessageItem(hello);
        salutation.setRect(100,10,100,10);
        mainwindow.addItem(salutation);
        ItemString inputstr =
        new ItemString("Get input...");
        ButtonItem inputButton = new ButtonItem();              //←④
        inputButton.setButtonShape(ButtonItem.BTN_ZAB);
        inputButton.setRect(100,100,100,100);
        inputButton.setWink(true);
        inputButton.addChangeListener(
            new GWChangeListener()
            {
                public void stateChanged(GWEvent e)
                {
                    String answer =
                        SoftKeyboardWindow.request(
                            "Add your Input",
                            "", 256,
                            ItemString.LANG_ENGLISH_US
                        );
                    inputdisplay.setMessage(
                            new ItemString(answer));
                    inputdisplay.paint();
                }
            }
        );
        mainwindow.addItem(inputButton);
        ItemString label1 =
            new ItemString("Your Input Was...");                //←⑤
        MessageItem label = new MessageItem(label1);
        label.setRect(100,110,200,10);
        mainwindow.addItem(label);
        ItemString outputstr = new ItemString ("");
        MessageItem
        inputdisplay = new MessageItem(outputstr);              //←⑥
        inputdisplay.setRect(100,125,200,10);
        mainwindow.addItem(inputdisplay);
    }
}
```

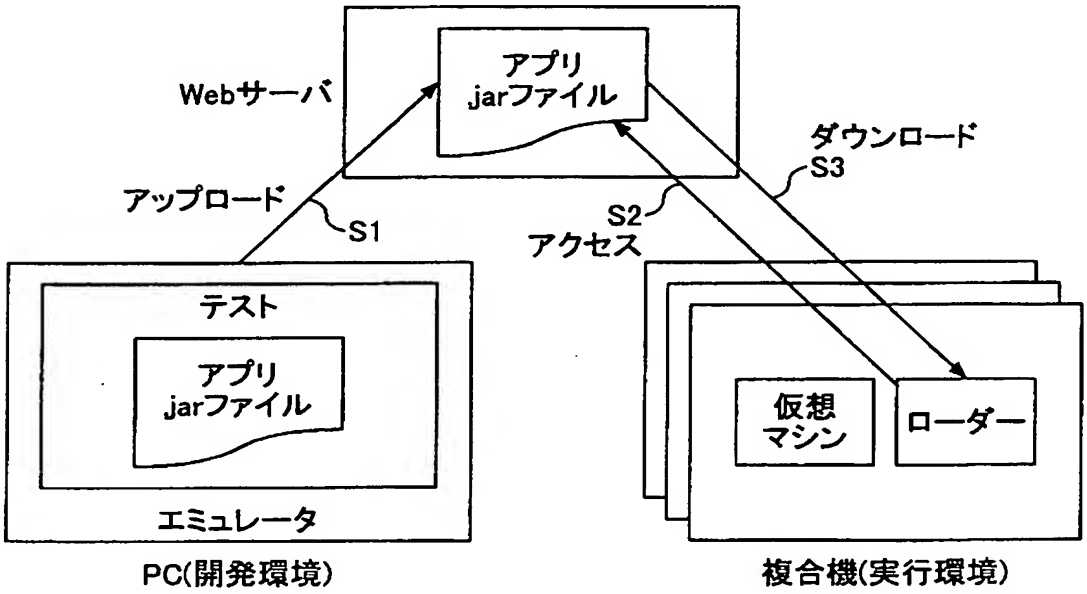
【図３７】

サンプルプログラムの実行結果を示す図



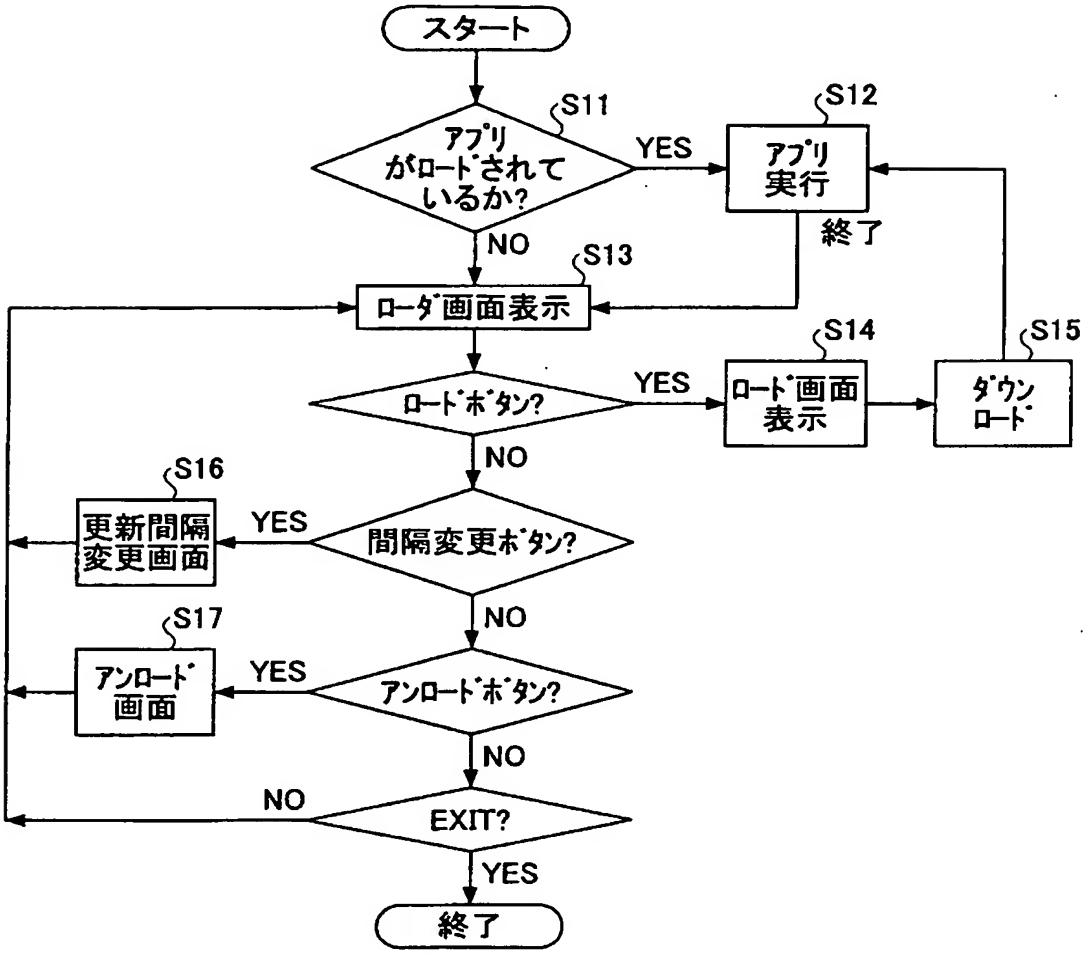
【図３８】

図２９に示した構成におけるＪａｖａアプリの
アップロードからダウンロードまでの概念図



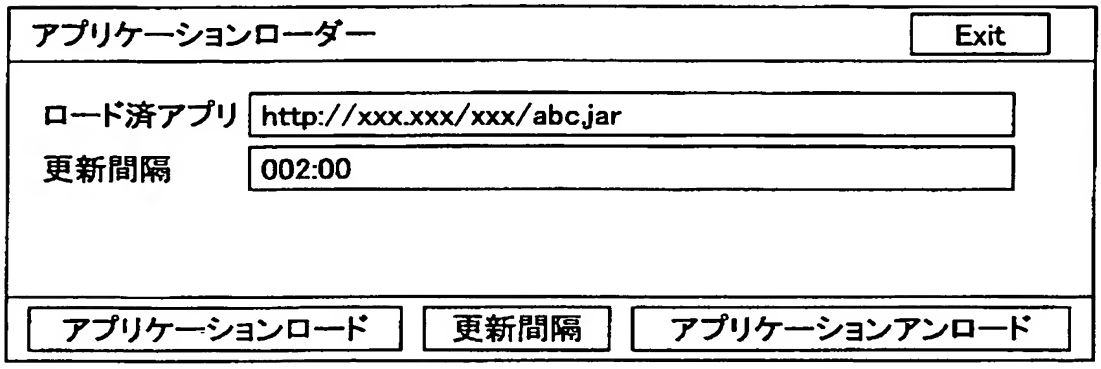
【図３９】

ローダーの処理手順を示すフローチャート



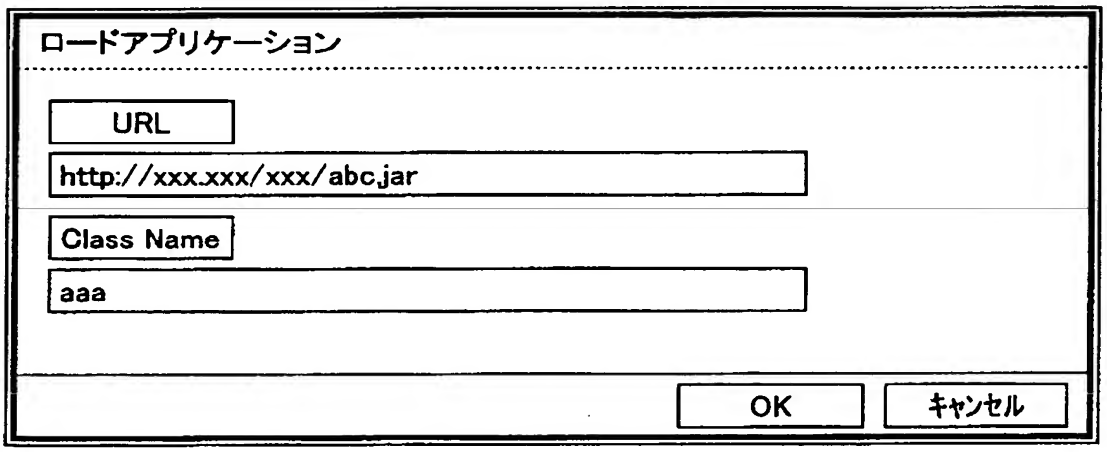
【図４０】

複合機に表示されるローダーの画面を示す図



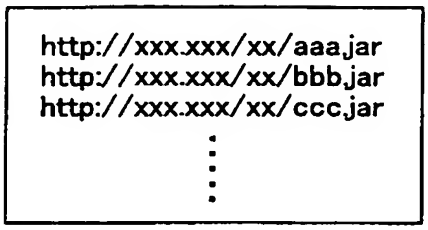
【図４１】

アプリをロードするための画面



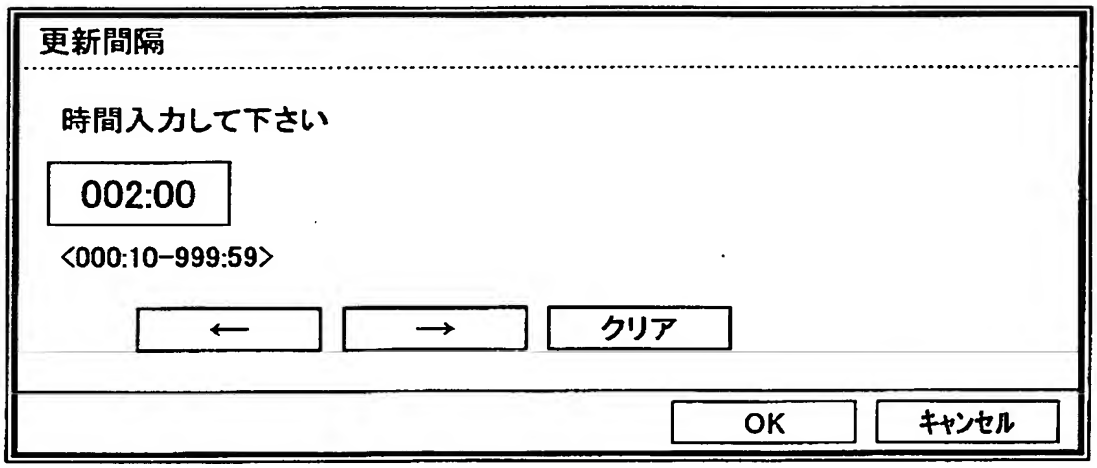
【図４２】

ＪａｖａアプリのＵＲＬのリスト



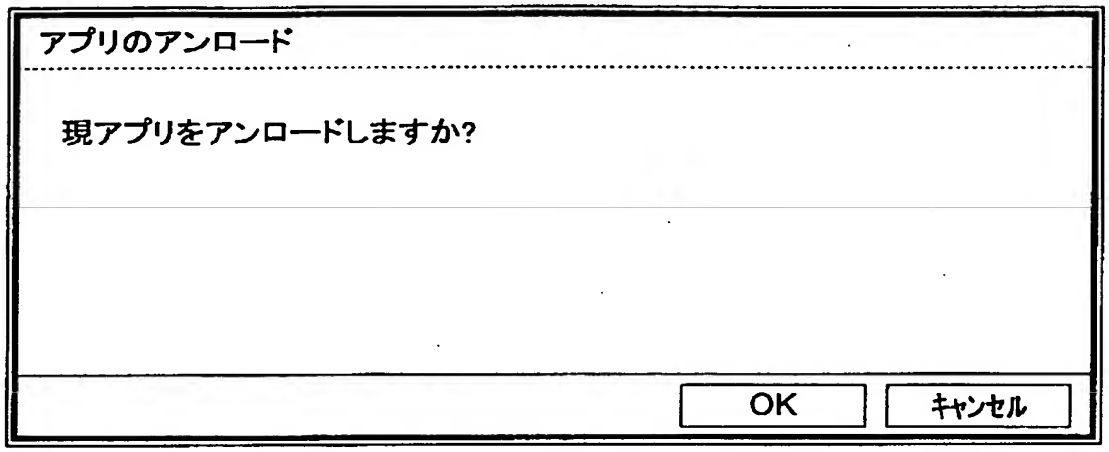
【図４３】

更新間隔を変更するための画面



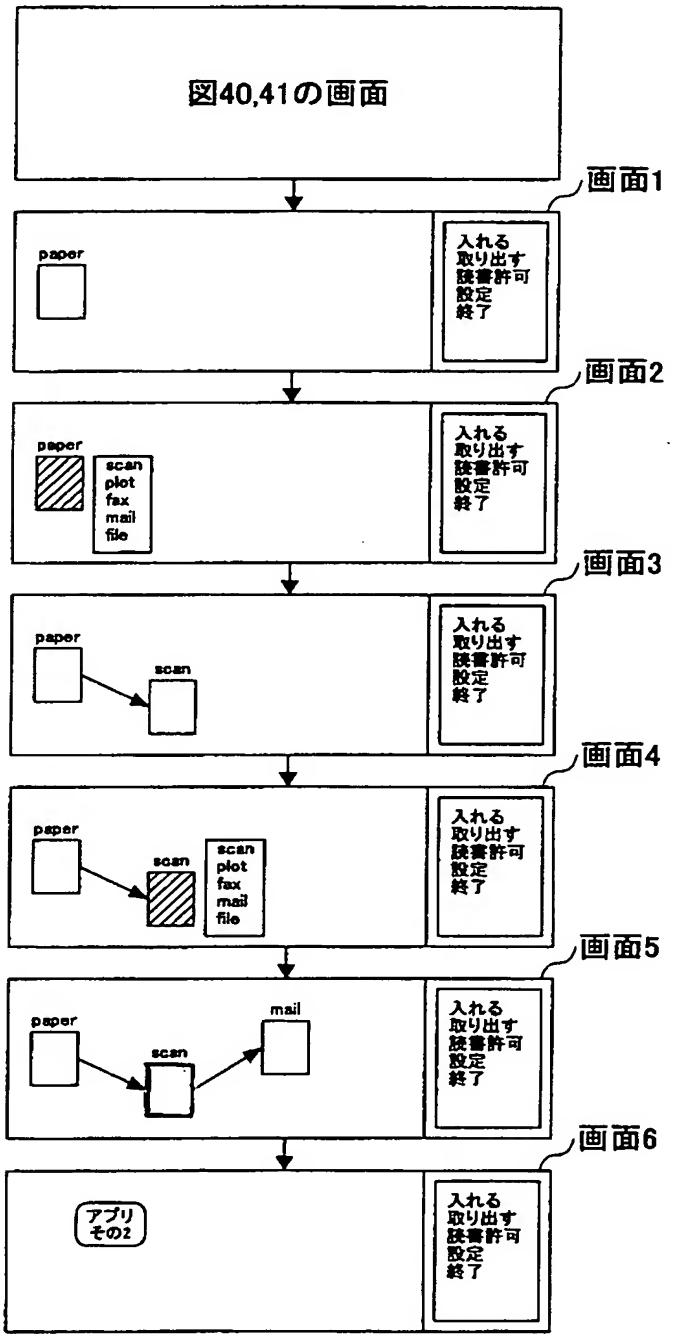
【図４４】

アプリをアンロードするための画面



【図４５】

Javaアプリを連結することにより、
カスタマイズプログラムを作成する手順を説明するための図



【書類名】　　要約書

【要約】

【課題】　アプリケーションを容易に実行可能な画像形成装置およびアプリケーション実行方法を得る。

【解決手段】画像形成処理に関するシステム側の処理を行うサービスモジュールを有し、当該サービスモジュールとは別にアプリケーションを搭載可能に構成された画像形成装置に、アプリケーションを実行する仮想マシンと、当該仮想マシンにより実行されるアプリケーションを管理するアプリケーション管理部とを備える。

【選択図】　　図３８

特願２００３－１９９９４７

出　願　人　履　歴　情　報

識別番号　　［０００００６７４７］

１．変更年月日　　１９９０年　８月２４日
［変更理由］　　新規登録
住　所　　東京都大田区中馬込１丁目３番６号
氏　名　　株式会社リコー

２．変更年月日　　２００２年　５月１７日
［変更理由］　　住所変更
住　所　　東京都大田区中馬込１丁目３番６号
氏　名　　株式会社リコー